Pareto Task Assignments by an Adaptive Quantum-based Evolutionary Algorithm AQMEA

Jerzy M. Balicki[1]

***Abstract*—** Pareto distributed task assignments found by the AQMEA (Adaptive Quantum-based Multi-criterion Evolutionary Algorithm) have been considered for computer systems. A quantum chromosome representation with the registry of the smallest units of quantum information has been applied. Evolutionary computing with Q-bit chromosomes has been proofed to ensure the enhanced population diversity than other representations, since individuals represent a linear superposition of states probabilistically. Moreover, the multi-criterion task assignment problem has been considered.

***Keywords*—** quantum simulation, evolutionary algorithms, distributed systems, multi-criterion optimization, artificial intelligence.

I. INTRODUCTION

Foundations of infinite dimensional ordered spaces create the base for decision making among several criteria that has been stated by Cantor and Hausdorff [23]. In complex decision situation, some goals, criteria or players are in conflict that is a subject of theory of games in the context of a social exchange economy formulated by von Neumann and Morgenstern [28]. Multi-criteria optimization and also game situations are tidy connected. Some Pareto solutions can be treated as results obtained by minimax strategies under some conditions [1].

Several artificial intelligence meta-heuristics can be applied to solve some multi-objective optimization problems [5]. Particle swarm optimization (PSO) integrates swarming behaviors detected in flocks of birds, schools of fish, swarms of bees or ants, and even human social behavior, from which the swarm intelligence paradigm has appeared. The significant strong point of PSO is its fast convergence among many global optimization algorithms.

Similarly to PSO, genetic algorithms operate on the population of some individuals [6]. However, in the genetic algorithm a fitness function is the transformation result of the problem preferences. Moreover, some procedures like mutation, crossover, and inversion are applied to exchange information between individuals.

For artificial immunological systems some approaches are based on proceeding the set of solution, currently. For instance, in a negative selection algorithm the set of antigens and the set of antibodies is considered. The measure of similarity between selected antigen and antibodies plays the role of criterion for increasing the fitness of an antibody [6].

In artificial neural networks, set of neurons is trained to calculate a response that is adequate to the input data [22]. However, one solution is produced by a neural network, only. So, we can use an evolutionary algorithm to operate the population of artificial neural networks.

The ant colony algorithm is an interesting paradigm that considers the set of artificial ants to calculate one solution to the problem [12]. Ants cross the edges of the graph and mark their paths by the amount of pheromone. The shorter path to the goal, the more intensive amount of pheromone. Moreover, amount of pheromone decreases with the next step, currently. An ant selects the edge with the probability proportional to the amount of pheromones that characterized the edge.

Fuzzy logic algorithms, simulated annealing, tabu search, and the Boltzmann machines are the other crucial paradigms for computer decision making due to distributed systems [29].

A quite different approach is a quantum-inspired algorithm that can be used for a computer decision aid, too. It is possible to construct a microscopic quantum mechanical Hamiltonian model of computer based on Turing machine [7]. Some premature simulation models of physics for computer implementations have been studied by Feynman [13]. Then, crucial principles of quantum theory have been established by Deutsch [8]. Furthermore, the Church-Turing principle has been verified due to the universal quantum computer that is based on a quantum Turing machine [9].

Furthermore, some quantum computational networks have been studied. It is an original distributed system that can have much higher calculation power standard digital distributed systems and quantum computers.

Quantum computers accelerate the efficiency of processing, but they do not permit computing functions that are not theoretically computable by classical computers due to the Church-Turing thesis: "*Every function which would naturally be regarded as computable can be computed by the universal Turing machine*" [9].

[1] Manuscript was received December 11, 2010. This work was supported in part by the Ministry of Science and High Education of Poland under Grant OR00010811.

Jerzy M. Balicki is with the Faculty of Electronics, Telecommunications and Informatics, Gdansk University of Technology 11/12 Gabriela Narutowicza Street, 80-233 Gdańsk, Poland (e-mail: Balicki@eti.pg.gda.pl).

Quantum algorithms for the factoring problem have been constructed by Shor [24]. In this dilemma, it is given a composite number γ like 4, 6, 8, 9, …, and we find an integer p, strictly between 1 and γ, that divides γ. A reduction of the factoring problem can be done to the dilemma of order-finding that may be implemented on a digital computer. Then, a quantum algorithm can be applied to decipher the order-finding problem. Shor's algorithm is exponentially faster than the most efficient classical factoring algorithm. It can be applied to solve the public-key cryptography method RSA that is based on the assumption that factoring large numbers is computationally infeasible for classical computers. No algorithm is known that can find a factor in polynomial time. However, a quantum-based approach shows that the factoring could be efficient on a quantum computer. It is a motivator for the development quantum computers and quantum algorithms.

In this paper, we consider decision making by the AQMEA – the Adaptive Quantum-based Multi-criterion Evolutionary Algorithm [2] – for distributed computer systems. AQMEA operates on a population of Q-chromosomes. Evolutionary computing with Q-bit chromosomes ensures the population diversity better than other representations, since individuals represent a linear superposition of states probabilistically. Moreover, the multi-criterion problem for task assignment is considered due to the distributed system area.

II. Quantum-based Algorithms

An implementation of the quantum factoring algorithm with using a nuclear magnetic resonance confirmed that quantum computers can find some factor integers in polynomial time [27]. An implementation of the simplest instance for the factorization of $\gamma = 15$ was carried out by development seven spin -1/2 nuclei in a molecule as quantum bits. Q-bits can be manipulated with room temperature liquid-state nuclear magnetic resonance techniques. Using nuclei to store quantum data is scalable to systems with several Q-bits. Experimental and theoretical techniques were demonstrated for a precise control and modeling of complex quantum computers. Finally, a parameter-free, predictive model of de-coherence effects in system were prepared [27].

Rules of quantum computing by development quantum phenomena, such as an entanglement or superposition can be applied to carry out operations on Q-data [18]. Quantum entanglement is a property of the quantum mechanical state of a system containing two or more objects. These objects make up the system and they are linked in such a way that one cannot adequately describe the quantum state of any member of the system without full mention of the other members of the system, even if the individual objects are spatially separated. Quantum entanglement is at the heart of the EPR paradox that was developed by Albert Einstein, Boris Podolsky, and Nathan Rosen in 1935, and it was experimentally verified for the first time in 1972 by Stuart Freedman and John Clauser.

A genetic quantum algorithm and its application to combinatorial optimization problem can be considered [14]. In a quantum-inspired evolutionary algorithm (QEA), Q-bit chromosome is defined by a string of Q-bits that represents a linear superposition of binary states in search space probabilistically. The Q-bit individual ensures higher population diversity than other known representations.

Even QEA is based on the idea of quantum computing, it is not a quantum algorithm, but a digital evolutionary algorithm that simulates the quantum phenomena. To demonstrate its numerical performance, experiments on the knapsack problem have been carried out [14]. A quantum-inspired algorithm executed well devoid of premature convergence.

A Q-gate is a variation operator that drives the chromosomes toward better solutions and toward a single state [15]. A quantum gate is a basic quantum circuit operating on a small number of qubits. They are the building blocks of quantum circuits. Contrasting many digital logic gates, quantum logic gates are reversible. However, digital computing can be performed using only reversible gates. For instance, the reversible Toffoli gate can implement all Boolean functions. This gate has a direct quantum equivalent, showing that quantum circuits can perform all operations performed by classical circuits.

Quantum logic gates are represented by unitary matrices. The most common quantum gates operate on spaces of one or two qubits, just like the common classical logic gates operate on one or two bits. This means that as matrices, quantum gates can be described by 2×2 or 4×4 unitary matrices. A gate which acts on M qubits is represented by a $2^M \times 2^M$ unitary matrix. The number of qubits in the input and output of the gate have to be equal. The action of the quantum gate is found by multiplying the matrix representing the gate with the vector which represents the quantum state.

Firstly, there are various solutions represented probabilistically because a Q-bit register symbolizes the linear superposition of all possible states with the same probability. As the probability of each qubit converges either to 1 or 0 by the Q-gate, the Q-chromosome approaches to a single state and the diversity disappears progressively.

A parallel quantum-inspired genetic algorithm was constructed for combinatorial optimization problems, too [17]. The population of Q-individuals is divided on sub-populations performed by parallel set of quantum processors.

Some results related to setting the parameters of QEA have been studied for practical applications for combinatorial optimization [16]. That algorithm has been developed for the face verification [19] and for solving the Travelling Salesman Problem [25].

Recently, the QMEA quantum-inspired multi-objective evolutionary algorithm has been proposed for multiobjective 0/1 knapsack problems [20]. Experimental results showed that QMEA finds solutions close to the Pareto-optimal front while maintaining a better spread of non-dominated set. Another version of QMEA has been applied for a image segmentation [26]. Balicki proposed the other construction of QMEA for task assignment in computer networks [2]. Finally, an adaptive quantum-based multi-objective evolutionary algorithm (AQMEA) has been constructed [3]. Some QMEAs have been proved to calculate better outcomes than digital genetic algorithms.

To improve the quality of the non-dominated set as well as the diversity of population in multi-objective problems, an algorithm is proposed by employing the concept and principles of quantum computing such as uncertainty, superposition, and interference. We extend the AQMEA to improve proximity to the Pareto-optimal front, preserving diversity intact by employing advantages of quantum-inspired evolutionary algorithm. The improving proximity means to find the better solutions which are evaluated as good individuals by fitness function.

In some multi-objective evolutionary algorithms is applied a strong elitist method with a procedure to maintain diversity efficiently using non-dominated sorting and crowding distance assignment [26]. It is even more powerful if the elitism is further strengthened and the solutions are spread out by quantum mechanism. Multiple observations of Q-bit individuals allow a local search in the area of the non-dominated solutions.

Also, maintaining best Q-bit individuals in every generation can avoid the possibility of losing high quality individuals. Furthermore to deal with quantum computing concepts, the comparison mechanism is presented between the best group and the others. Convergence and preservation of diversity being the key issues under scrutiny, the proposed approach is expected to help improve the performance of AQMEA.

III. Adaptive Quantum-based Multiobjective Evolutionary Algorithm

An adaptive quantum-based multiobjective algorithm AQMEA utilizes the probabilistic representation that is based on the concept of qubits [3]. A qubit is a two-layer quantum system that can be modeled as the Hilbert space H_2 with the given base $B = \{|0\rangle, |1\rangle\}$. The Bloch sphere is a geometrical demonstration of the state space for a qubit and it may also refer to the space of an *M*-level quantum system.

A Hilbert space *H* is a real or complex inner product space that is also a complete metric space with respect to the distance function induced by the inner product. It means that *H* is a complex vector space on which there is an inner product associating a complex number to each pair of elements of *H* that satisfies the properties of the complex conjugate, linearity, and positive definite. A real inner product space can be defined in the same way, apart from that *H* is a real vector space and the inner product takes real values.

A qubit may be in the "1" binary state, in the "0" state, or in any superposition of the two [10]. The state x_m of the *mth* qubit in the *Q*-chromosome can be represented, as follow:

$$Q_m = \alpha_m |0\rangle \oplus \beta_m |0\rangle, \qquad (1)$$

where

α_m and β_m - complex numbers that specify the probability amplitudes of the corresponding states,

$\oplus$ - a superposition operation,

m – the index of the gene in the chromosome, $m = \overline{1, M}$.

Value $|\alpha_m|^2$ is interpreted as the probability that we observe the state "0" . Similarly, $|\beta_m|^2$ is the probability that state "1" is observed. A qubit may be characterized by the pair (α_m, β_m). There is, as below [10]:

$$|\alpha_m|^2 + |\beta_m|^2 = 1. \qquad (2)$$

Let it be considered a four-bit register of a digital computer with 16 different four-bit strings 0000, 0001, 0010, …, 1110, 1111. If it is a deterministic computer, then a four-bit register is in one of those states with probability 1.

However, if it is a probabilistic computer, then there is a possibility of it being in any one of several different states. We can describe this probabilistic state by sixteen probabilities p_0, p_1,…, p_E, p_F . It means that a probabilistic computer is in one state from all possible states. There is a constraint that sum of these probabilities is equal to 1.

The state of a four-qubit quantum computer is described by a sixteen-dimensional vector (α_0, α_1,…, α_F) that is a result of a wave-function. However, the sum of the squares of the coefficient magnitudes, $|\alpha_0|^2 + |\alpha_1|^2 + ... + |\alpha_E|^2$, must be equal to one. Moreover, these coefficients are complex numbers.

Since states are represented by complex wave-functions, two states being added together will undergo interference. This is a main difference between quantum computing and probabilistic classical computing [11].

If we measure the four qubits, then we observe a four-bit string. The probability of measuring a string is equal the squared magnitude of that string's coefficients. Probability that we read state as 0000 is $|\alpha_0|^2$, probability that we read state as 0001 is $|\alpha_1|^2$, and probability that we read state as 1111 is $|\alpha_F|^2$. Thus a measurement of the quantum state with some complex coefficients (α_0, α_1,…, α_F) gives the classical probability distribution ($|\alpha_0|^2$, $|\alpha_1|^2$,..., $|\alpha_F|^2$). We say that the quantum state "collapses" to a classical state.

In linear algebra, *a basis* plays important role and it is a set of vectors that, in a linear combination, can represent every vector in a given vector space, and such that no element of the set can be represented as a linear combination of the others. A basis is a linearly independent spanning tree. A sixteen-dimensional vector can be specified in many different ways, depending on a basis chosen for the space. The basis of four-bit strings is known as the computational basis, and it is suitable. However, the other bases of unit-length, orthogonal vectors can also be applied [11].

Bra-ket (Dirac) notation is used for describing quantum states by development of angle brackets and vertical bars. Moreover, it is less common used in mathematics because the inner product (or dot product) of two states can be denoted by a *bracket*, $\langle \alpha | \beta \rangle$ consisting of a left part, $\langle \alpha$, called the *bra*, and a right part, $|\beta\rangle$, called the *ket* [10].

Dirac notation is often used to make the choice of basis. For example, the state (α_0, α_1,…, α_F) in the computational basis can be written, as below:

$\alpha_0|0000\rangle + \alpha_1|0001\rangle + \alpha_2|0010\rangle + ... + \alpha_E|1110\rangle + \alpha_F|1111\rangle$.

We apply the notation

$$|0001\rangle = (0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0).$$

The computational basis for a single qubit (two dimensions) is $|0\rangle$ = (1,0) and $|1\rangle$ = (0,1).

An alternative common basis consists of the eigenvectors of the Pauli-x operator: $|+\rangle = \frac{1}{\sqrt{2}}(1,1)$ and $|-\rangle = \frac{1}{\sqrt{2}}(1,-1)$.

So, for a classical state of M bits, a 2^M-dimensional probability distribution requires an exponential number of real numbers. We can think of the system as being one of the M-bit strings, but we do not know which one.

However, in quantum computing all 2^M complex coefficients need to be kept track of to see how the quantum computer calculates. For example, a 32-qubit quantum computer has a state described by 2^{32} complex numbers.

IV. Quantum chromosome

The Q-chromosome can be represented by the chromosome matrix, as follows [21]:

$$Q = \begin{bmatrix} \alpha_1 & \dots & \alpha_m & \dots & \alpha_M \\ \beta_1 & \dots & \beta_m & \dots & \beta_M \end{bmatrix} \qquad (3)$$

Moreover, the procedure of random selection of decision values is involved with a chromosome matrix. If the decision variable x_m is characterized by (α_m, β_m), then it is equal to 0 with the probability $|\alpha_m|^2$ and it is equal to 1 with $|\beta_m|^2$ [18].

AQMEA is working on a digital computer and collapsing into a single state does not occur in AQMEA. So, we simulate this process of observation.

A standard M-bit state and a quantum M-qubit state are 2^M-dimensional elements that are processed differently for standard and quantum computation.

In randomized computation, the application of stochastic matrices preserves that the sum of probabilities has to be equal to one. On the other hand, in quantum computation, allowed operations are unitary matrices. Those matrices preserve that the sum of the squares is equal to one [18].

A unitary matrix is an M by M complex matrix U satisfying the condition [18]:

$$U^* U = U U^* = I \qquad (4)$$

where

U^* - the conjugate transpose (the Hermitian adjoint) of U,

I - the identity matrix in M dimensions.

What sort of unitaries can be applied depends on the quantum devices.

Quantum computations are reversible because they are probabilistic combinations of unitaries. So, quantum computation generalizes classical computation.

Upon termination of the algorithm, the result needs to be read off. In the case of a classical computer, we sample from the probability distribution on the M-bit register to obtain one definite M-bit string, e.g. 0100100 for M=7.

In quantum computation, we *measure* the M-qubit state, which is equivalent to collapsing the quantum state down to a classical distribution with the coefficients being the squared magnitudes of the coefficients for the quantum state. It is followed by sampling from that distribution. However, this destroys the original quantum state.

Many algorithms only give the correct answer with a certain probability; however by repeatedly initializing, running and measuring the quantum computer, the probability of getting the correct answer can be increased.

The binary chromosome $x=(x_1,\dots,x_m,\dots,x_M)$ can be measured with the probability $p(x)$ calculated, as follows:

$$p(x) = \prod_{m=1}^{M} p(x_m), \qquad (5)$$

where

$$p(x_m) = \begin{cases} \alpha_m{}^2 & for\ x_m = 0 \\ \beta_m{}^2 & for\ x_m = 1 \end{cases}$$

Let us consider the chromosome, as below:

$$Q = \begin{bmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{5}} & \frac{2}{\sqrt{7}} & \frac{1}{\sqrt{2}} & \frac{1}{2} \\ \frac{-\sqrt{2}}{\sqrt{3}} & \frac{2}{\sqrt{5}} & \frac{\sqrt{3}}{\sqrt{7}} & \frac{1}{\sqrt{2}} & \frac{\sqrt{3}}{2} \end{bmatrix}$$

It means that the binary chromosome (0,0,0,0,0) can be observed with the probability 1/210, the binary chromosome (0,0,0,0,1) – 1/70, and so on. The sum of all above probabilities is equal to 1.

Evolutionary computing with Q-bit representation has a better population diversity than other representations, since it can represent linear superposition of states probabilistically. Although, the search space Ω for an evolutionary algorithm consists of 2^M elements, this space can be generated by one Q-bit chromosome.

There is, as below:

$$\sum_{x \in \Omega} p(x) = 1. \qquad (6)$$

AQMEA is a probabilistic algorithm similar to other evolutionary algorithms. However, it maintains an additional population of Q-chromosomes. Although, the search space is represented by a Q-chromosome, we consider the population of ζ quantum-chromosomes because of destroying the measured values. In fact, only one value of Q-chromosome can be measured and the others are supposed to be missed.

We model this situation by the ζ quantum-chromosomes based on complex numbers α_m and β_m , and formulas (2), (3).

An initial population can be generated by random numbers $|\alpha_m| \in [-1;1]$ and calculation $|\beta_m|$ from (2). After formulation the first *Q*-chromosome, the next one is formed until the whole population of ζ quantum chromosomes is created.

Then, we observe states of *Q*-chromosomes. A values of genes in the generated *x*-chromosome are selected either 0 or 1 for each bit using the probabilities from the related *Q*-chromosome. In a quantum computer, during the observing a quantum state, it collapses to a single state. So, an observation of the quantum chromosome state products one state of a binary *x*-chromosome. However, collapsing into a single state does not occur in QEA automatically, since it is working on a digital computer, not a quantum computer. So, we simulate the process of observation for quantum chromosomes.

We observe the *Q*-chromosome from the quantum population $\Theta(t)$, where *t* is the number of population ($t \le T_{\max}$). We observe the same chromosome λ times. The parameter of sampling should be significantly smaller than size of the search space 2^M, of course. In results, λ digital *x*-chromosomes are produced from one *Q*-chromosome. If the number of *Q*-individuals is equal to ζ, the size of a binary population $L=\lambda\zeta$.

Afterwards, we evaluate the fitness for each binary chromosome by using the ranking procedure [2]. We determine non-dominated solutions from the current population and copy them to the archive.

Quantum approach for implementation of genetic algorithms has several advantages that are similar to the quantum applications.

Integer factorization is computationally non-admissible by a digital computer for large integers that are the product of only a few prime numbers. On the other hand, a quantum computer could efficiently solve this problem using Shor's algorithm to find its factors. This ability would allow to "crack" many of the cryptographic systems, because there are a polynomial time algorithm for solving the problem. In particular, most of the public key ciphers are based on the difficulty of factoring integers or the related discrete logarithm problem, e.g. RSA. These are used to protect secure Web pages or encrypted email. It is a major result for electronic privacy and security. The approach to increase the security of an algorithm like RSA rely on increasing the key size to such amount that an enemy does not have a technology to build and use a powerful enough q-computer. However, it is possible to obtain the powerful quantum technology in the next decade.

More advanced approach is base on implementation quantum cryptography. There are some digital signature schemes that may be secured against quantum computers, e.g. Lamport signatures [18].

Quantum algorithms give not only polynomial speedup over the classical procedures, including the simulation of quantum physical processes from chemistry and solid state physics, the approximation of Jones polynomials, and solving Pell's equation [17]. For some questions, quantum algorithms give a polynomial speedup. An example is quantum database search that can be solved by Grover's algorithm using quadratically fewer queries to the database than by standard computers. Several other examples of provable quantum speedups for query problems are the finding collisions in two-to-one functions and evaluating NAND trees [25].

V. Evolutionary operations

In Adaptive Quantum-based Multi-objective Evolutionary Algorithm AQMEA, there are two populations. The first one is a *Q*-population Θ that consists on ζ quantum-individuals, and the second one is a digital *x*-population *P* that consists on *L* individuals. Moreover, each *Q*-chromosome generates λ *x*-chromosomes, randomly with the given probabilities.

We could not calculate the fitness of the quantum chromosome, directly. Fitness of quantum chromosome is necessary for carried out the selection and crossover. However, the quantum chromosome can be characterized by the selection probability $p_s(Q_i)$, as follows:

$$p_s(Q^i) = \frac{\sum_{j=1}^{\lambda} f(x^{ij})}{\sum_{x^{ij} \in P(t)} f(x^{ij})}, \quad i = \overline{1,\zeta} \tag{7}$$

where

Q^i – the *i*th quantum chromosome from the current *Q*-population $\Theta(t)$,

x^{ij} – the *j*th binary chromosome determined randomly from the *i*th quantum chromosome; x^{ij} belongs to the current *x*-population *P(t)*.

We can prove, as follows:

$$\sum_{i=1}^{\zeta} p_s(Q^i) = 1 \tag{8}$$

A quantum crossover is based on the cross-overing of two matrixes given by formula (3). Each of matrixes is supposed to represent the *Q*-individual selected from the current quantum population with the probability p_s.

A short overview of classical evolutionary algorithms for multi-objective optimization problems is submitted in [4, 20, 26]. The name "adaptive evolutionary algorithm" for evolutionary algorithms is related to the changing of some parameters as a crossover probability, a mutation rate, and a population size during the evolutionary searching [5].

The crossover probability is decreased due to the number of new generations, as follows:

$$p_c = e^{-\xi t / T_{\max}} \tag{9}$$

where

e – the Euler constant,

ξ – crossover rate parameter, usually $\xi = 4$.

Figure 1 shows the decreasing values of crossover rate p_c due to the increasing the generation number *t*, for $\xi = 4$, $T_{\max}=200$. At the beginning of evolutionary calculations, the crossover rate decreases: 0.9802, 0.9608, 0.9418, … . It means

that almost all individuals are selected to cross-overing. In the middle steps of evolutionary computations when t=100, the rate has got much smaller results and decreases slower: 0.1275, 0.1249, 0.1225, … . In this period of time, about twelve percent of population is taken to the generation of the offspring pair. Finally, when t approaches to $T_{\max}$, a crossover rate is equal to 0.0191, 0.0187, and 0.0183. In this stage, crossover plays the small role because the good areas should be found and we need rather local optimization by mutation or an additional procedure.

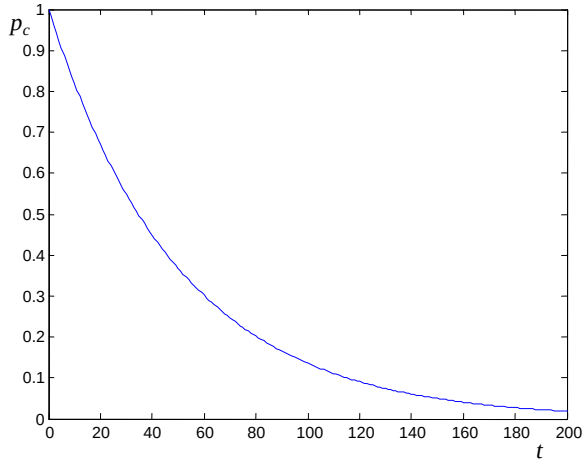

Fig. 1. The decreasing values of crossover rate p_c due to the increasing the generation number t, for ξ =4, $T_{\max}$=200.

Figure 2 shows a scheme of the adaptive quantum-based multi-criterion evolutionary algorithm AQMEA. We randomly generate ζ quantum-chromosomes for the quantum population $\Theta(0)$, where $\zeta \le 10$. Because the quantum algorithm is simulated on the digital computer, the structure of Q-chromosome is determined by (3) subject to (2). An initial population $\Theta(0)$ can be generated by random numbers $|\alpha_m| \in [-1;1]$ and calculation $|\beta_m|$ from (2).

Then, we produce an initial binary population $P(0)$ that consists od L x-chromosomes generated λ times from each quantum-chromosome. If the gene x_m is characterized by (α_m, β_m), then it is equal to 0 with the probability $|\alpha_m|^2$ and it is equal to 1 with $|\beta_m|^2$.

The binary search space consists on 2^M elements that are represented by x-chromosomes. If x is admissible, then the fitness function value is estimated, as below:

$$f(x) = r_{\max} - r(x) + P_{\max} + 1, \qquad (10)$$

where

$r(x)$ – the rank of an admissible solution, $1 \le r(x) \le r_{\max}$,

$P_{\max}$ – the maximal value of the penalty function defined for non-admissible solutions.

In the quantum two-weight tournament selection (Fig. 2, line 14), the roulette rule is carried out twice due to $p_s(Q^i)$ for each Q^i from the current quantum population. If two potential Q-parents (a, b) are selected, we observe a Q-parent λ times and produce λ binary chromosomes. Then, we determine $N(a)$ - the set of non-dominated solutions from binary chromosomes generated for the quantum individual a. Similarly, we obtain $N(b)$. There is possible that one solution is included to the $N(a)$ or $N(b)$. In fact, it could be for the superior solution that dominates the others from the temporary set.

1. BEGIN
2. t:=0, t – the number of population
3. set ζ the size of Q-population Θ, L size of binary population $\boldsymbol{P}$, $L = \lambda\zeta$, for the given sampling parameter λ
4. p_m:=1/(M ζ), M – the length of x
5. generate an initial population $\Theta(0)$ *and* $\boldsymbol{P}(0)$,
6. calculate non-dominated ranks $r(x)$ and fitness $f(x)$, $x \in \boldsymbol{P}(t)$
7. *finish*:=FALSE
8. WHILE NOT *finish* DO
9. BEGIN /* new populations Θ and $\boldsymbol{P}$ */
10. t:= t+1, $\boldsymbol{P}(t) := \emptyset$, $\Theta(t) := \emptyset$
11. calculate the selection probabilities $p_s(x)$, $x \in \boldsymbol{P}(t-1)$ by (6)
12. FOR ζ /2 DO
13. BEGIN /* reproduction cycle */
14. 2WT-selection of a potential parent pair {***a,b***} from Θ(*t*-1)
15. Q-crossover of a parent pair {***a,b***} with the adaptive crossover rate p_c, $p_c := e^{-\xi t / T_{\max}}$
16. Q-mutation of an offspring pair {***a',b'***} with the adaptive mutation rate, $p_m := \dfrac{t}{\zeta M T_{\max}}$
17. Θ (*t*):= Θ(*t*)∪{***a',b'***}
18. END
19. generate $\boldsymbol{P}(t)$ by observing $\boldsymbol{\Theta}(t)$ λ times
20. calculate ranks $r(x)$ and fitness $f(x)$, $x \in \boldsymbol{P}(t)$
21. IF ($\boldsymbol{P}(t)$ converges OR $t \ge T_{max}$) THEN *finish*:=TRUE
22. END
23. END

Fig. 2. An adaptive quantum-based multi-criteria evolutionary algorithm AQMEA

Afterwards, individuals from $N(a)$ and $N(b)$ are compared due to the Pareto relationship of domination. The dominated individuals are eliminated and $N(a,b)$ the new non-dominated Pareto solution set is created. If at least one solution from $N(a)$ belongs to $N(a,b)$, then the quantum chromosome a is accepted. The same rule is applied for the selection for cross-overing the Q-chromosome b.

If potential quantum parents a, b generate binary solutions that are non-admissible, then the alternative with the corresponding smaller penalty is selected. Then, the random selection is repeated.

The fitness sharing technique can be substituted by the adaptive changing of main parameters and the generation of binary solution by observing quantum individuals. The quality of attained solutions increases in optimization problems with

one criterion, if the crossover probability and the mutation rate are changed in an adaptive way.

The crossover point is randomly chosen for the quantum chromosome in the *Q*-crossover operator (Fig. 2, line 15). The crossover point is selected between two columns of the matrix (3) and separates it on two sub-matrixes. These sub-matrixes are exchanged with corresponded sub-matrixes from the other quantum chromosome. It is worth to notice that crossover is carried out on quantum chromosomes instead of binary chromosomes as at digital genetic algorithm.

The crossover probability decreases from 1 at the initial population and almost each pair of potential parents is obligatory taken for the crossover procedure. A crossover operation supports the finding of a high-quality solution area in the search space. It is important in the early search stage, especially. If the number of generation increases, the crossover probability decreases. Some search areas with the high quality solutions are identified after several crossover operations. That is why, value p_c decreases to 0.12, if t =100 for maximum number of population T_{max}=200. The final smallest value p_c is 0.02.

In *Q*-mutation (Fig. 2, line 16), the random swap of the column from matrix (3) by another one is applied. If the quantum gene $\begin{bmatrix}\alpha_m \\ \beta_m\end{bmatrix}$ is randomly taken for mutation, the new value α_m is randomly taken from the period [-1; 1]. Then, the value β_m is calculated from (2). A mutation rate increases according to the progress of the generation number.

To improve the quality of solution, we propose the development of the negative selection algorithm (NSA) from an immune systems. The immune system can be seen as a distributed adaptive system. The negative selection algorithm is based on the discrimination principle that is used to know what a part of the immune system is.

An antigen is a molecule that stimulates a response against trespassers. The term originated from the notion that they can stimulate antibody generation. Moreover, the immune system consists of some viruses as well as bacteria. An antibody (an immunoglobulin) is a large *Y*- shaped protein used to identify and neutralize foreign objects like bacteria and viruses. The antibody recognizes a specific target. The negative selection can be used to manage constraints in an evolutionary algorithm by isolating the contemporary population in two groups. Feasible solutions called "antigens" create the first cluster, and the second cluster of individuals consists of "antibodies" – infeasible solutions.

We assume the initial fitness for antibodies is equal to zero. Then, a randomly chosen antigen G^- is compared to the selected antibodies. After that, the distance *S* between G^- and the antibody B^- is calculated due to the amount of similarity at the genotype level. The measure of genotype similarity between the antigen and the antibody depends on their representation.

This assessment of similarity for the integer version is, as follows [3]:

$$S(G^-, B^-) = \sum_{m=1}^{M} \left| G_m^- - B_m^- \right|, \qquad (11)$$

where

M – the length of the solution,

G_m^- – value of the antigen at position m, $m = \overline{1, M}$,

B_m^- – value of the antibody at position m, $m = \overline{1, M}$;

The negative selection can be implemented by an external genetic algorithm to the AQMEA. In that approach, infeasible solutions that are similar to feasible ones are preferred in the current population. Although, almost all the random choices are based on the uniform distribution, the pressure is directed to improve the fitness of appropriate infeasible solutions.

VI. Quality of Solutions

Let the Pareto points $\{P_1, P_2, ..., P_U\}$ be given for the considered instance of the optimization problem with N criteria, and let points $\{A_1, A_2, ..., A_W\}$ be produced by an algorithm. The level of convergence to the Pareto front is calculated due to the Euclid distance, as follows:

$$S = \sum_{w=1}^{W} \sqrt{\sum_{n=1}^{N} \min_{u=1,U} \left(P_{un} - A_{wn}\right)^2}. \qquad (12)$$

An average arithmetic level $\overline{S}$ is calculated for several runs of the evolutionary algorithm and for set of *S* values..

As reported in [3], the best outcomes were obtained for some task assignment problems by the AMEA+. This algorithm gives better results than the previous AMEA. After 200 generations, an average level of Pareto set obtaining is 1.8% for the AMEA+, 3.4% for the AMEA. 30 test preliminary populations were prepared, and each algorithm starts 30 times from these populations. For integer constrained coding of chromosomes, there are 12 decision variables and the search space consists of 25 600 solutions.

For the other instance with 15 tasks, 4 nodes, and 5 computer sorts there are 80 binary decision variables. An average level of convergence to the Pareto set is 16.7% for the AMEA+ and 18.4% for the AMEA. A maximal level is 28.5% for the AMEA+ and 29.6% for the AMEA. For this instance the average number of optimal solutions is 19.5% for AMEA+ and 21.1% for AMEA.

An average level of convergence to the Pareto set, an maximal level, and the average number of optimal solutions become worse, when the number of task, number of nodes, and number of computer types increase. An average level is 34.6% for the AMEA+ versus 35,7% for the AMEA, if the instance includes 50 tasks, 4 nodes, 5 computer types and also 220 binary decision variables.

It is worth to mention that those calculations were carried out by a digital computer, but the expected results can be obtained for quantum computers. Consider a problem of a password cracker. In this dilemma, there is the only way to solve it by guessing answers repeatedly and check them.

Moreover, there are some possible answers to check, and also every possible answer takes the same amount of time to check. Finally, there are no clues about which answers might be better: generating possibilities randomly is just as good as checking them in some special order [10]. In a password cracker problem, there are attempts to guess the password for an encrypted file. We assume that the password has a maximum possible length.

For such problems, the time for a quantum computer to solve this is proportional to the square root of *n* (number of possible answers about password). That can be a very large speedup, reducing a calculation time from months to milliseconds. It can be used to attack symmetric ciphers such as Triple DES and AES by attempting to guess the secret key [11].

Grover's algorithm can also be used to obtain a quadratic speed-up over a brute-force search for a class of problems known as NP-complete [10]. Since chemistry and nanotechnology rely on understanding quantum systems, and such systems are impossible to simulate in an efficient manner classically, many believe quantum simulation will be one of the most important applications of quantum computing [11].

VII. Results for Benchmark Problem

To test the ability of the AQMEA, we consider a multi-criterion optimization problem for task assignment in a distributed computer system, where three criteria are optimized. In the formulated task assignment problem as a multi-criterion question, both $Z_{\max}$ – a workload of a bottleneck computer and C – the cost of system are minimized; in contrast, R – a reliability of the distributed system is maximized. Moreover, there are constraints for the performance of the distributed systems and the probability that all tasks meet their deadlines. In addition, constraints related to memory limits and computer locations are imposed on the feasible task assignment.

The first criterion is the workload of the bottleneck computer for the allocation x, and its values are provided by the subsequent formula [5]:

$$Z_{\max}(x)=\max_{i\in 1,I}\left\{\sum_{j=1}^{J}\sum_{v=1}^{V}t_{vj}x_{vi}^{m}x_{ij}^{\pi}+\sum_{v=1}^{V}\sum_{\substack{u=1\\u\neq v}}^{V}\sum_{\substack{i=1\\k\neq i}}^{I}\sum_{k=1}^{I}\tau_{vuik}\right. \quad (13)$$

where

$$x=\left(x_{11}^{m},...,x_{1I}^{m},...,x_{vi}^{m},...,x_{VI}^{m},x_{11}^{\pi},...,x_{1J}^{\pi},...,x_{ij}^{\pi},...,x_{IJ}^{\pi},N_1,...,N_v,...,N_V\right),$$

τ_{vuik} – the total communication time between the task T_v assigned to the ith node and the T_u assigned to the kth node.

Figure 3 shows three cuts in task assignment graph. We can balance workload among several processors by finding an optimal value of the bottleneck computer.

Let π_j be failed independently due to an exponential distribution with rate $\tilde{\lambda}_j$. We do not take into account of repair and recovery times for failed computer in assessing the logical correctness of an allocation. Instead, we are supposed to allocate tasks to computers on which failures are least likely to occur during the execution of tasks. Computers and tasks can be assigned to nodes in purpose to maximize the third criterion – the reliability function R defined, as below [4]:

$$R(x)=\prod_{v=1}^{V}\prod_{i=1}^{I}\prod_{j=1}^{J}\exp(-\tilde{\lambda}_j t_{vj}x_{vi}^{m}x_{ij}^{\pi}) \quad (14)$$

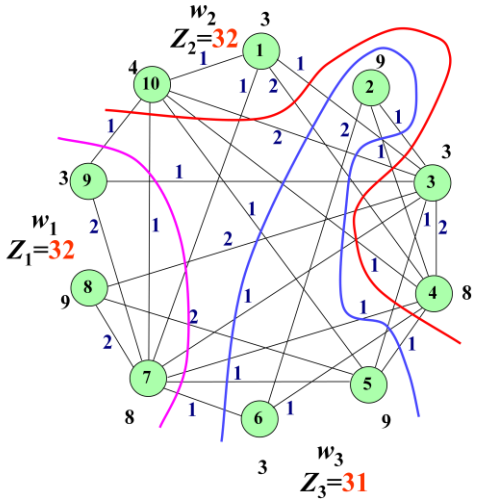

Fig. 3. Load balancing by finding an optimal task assignment

The second measure of the task assignment is a cost of computers that is calculated, as below:

$$C(x)=\sum_{i=1}^{I}\sum_{j=1}^{J}\kappa_j x_{ij}^{\pi} \quad (15)$$

where κ_j corresponds to the cost of the computer π_j.

The minimal performance of the distributed systems $\Xi_{\min}$ is supposed to be smaller than the performance of the entire system that can be estimated according to the following formula:

$$\Xi(x)=\sum_{i=1}^{I}\sum_{j=1}^{J}\delta_j x_{ij}^{\pi} \quad (16)$$

where δ_j is the numerical performance of the computer π_j for the task benchmark, for instance [MFlops].

The probability that all tasks meet their deadlines is supposed to be greater than the minimal probability $P_{\min}$. This parameter is usually set to be greater than 0.9.

$$P_D(x)=\sum_{l=1}^{K}p_l\prod_{m_v\in M_l}\xi(d_v-C_v(x)) \quad (17)$$

Two main constraint types: the benchmark performance limit and also probability that all tasks meet their deadlines are supposed to be complement with some resource constraint.

Figure 4 shows the cut of the evaluation space that is explored by the most effective meta-heuristic AMEA* [3]. Evolutionary algorithm AMEA* [3], the ant algorithm [12]

and genetic programming MGP [4] have been applied for solving a benchmark versions of multi-criterion task assignment. We can compare quality of obtained solutions by AQMEA to qualities produced by the other multi-criterion meta-heuristics.

The binary search space consisted of 1.0737×10^9 elements and included 25 600 admissible solutions. An average level $\overline{S}$ was calculated for fifty runs of each algorithm. After 350 assessments of those functions, an average level of Pareto set obtaining is 1.5% for the AQMEA, and 1.7% for the AMEA*. The other approaches gave much worse results.

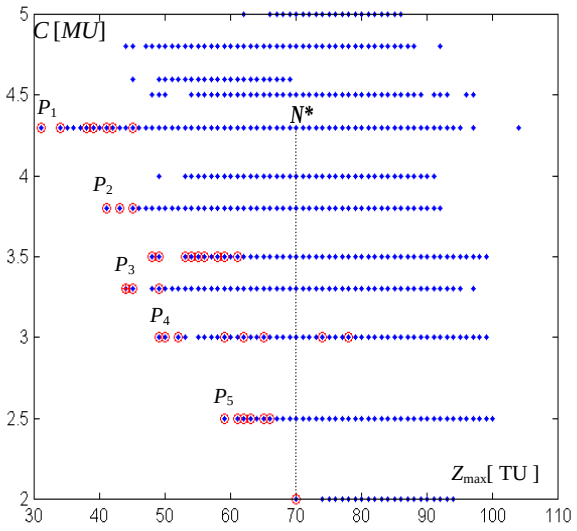

Fig. 4. Pareto-optimal task assignments obtained by AQMEA

Quantum search provides a promising alternative for the other problems like the face verification problem [25], disk allocation [21] or the Travelling Salesman Problem [25].

However, there are a number of practical difficulties in building a quantum computer, and quantum computers have only solved trivial problems. DiVincenzo listed some requirements for a practical quantum computer [11]. Quantum computers are supposed to be scalable physically to increase the number of qubits. Qubits should be initialized to arbitrary values and quantum gates are expected to perform faster than de-coherence time. What is more, a universal gate should be set, and qubits are supposed to be read easily.

A crucial question is controlling or removing de-coherence what is related to isolating the system from its environment as the smallest interaction with the external systems would cause the inner system to de-cohere. This result is irreversible, as it is non-unitary, and is generally something that is supposed to be avoided, if not highly controlled. The transverse relaxation (de-phasing) time for some technologies, typically range between nanoseconds and seconds at low temperature [11].

These implementations are more complicated for optical approaches when the timescales are orders of magnitude lower and an optical pulse shaping is applied. Error rates are usually proportional to the proportion of operating time to de-coherence time. So, an operation is required to be completed much more quickly than the de-coherence time [18].

If the error tempo is small enough, it is possible to apply quantum error correction, which corrects errors due to de-coherence. This is the constraint that the total calculation time should be longer than de-coherence period. This implies that a gate is supposed be able to execute its job 10^4 times faster than the de-coherence time of the computer [10].

The development of error correction is related to the increased number of required qubits. The number required to factor integers using Shor's algorithm is between n and n^2, where n is the number of bits in the number to be factored. Error correction procedure inflates this result by an additional factor of n. For n=100, it implies necessitate for about 10^3 qubits without error correction. With error correction, the figure would rise to about 10^5 qubits [11].

Another approach to the stability-decoherence problem is to create a topological quantum computer with quasi-particles used as threads and relying on braid theory to form stable logic gates [18].

In 2009, researchers at Yale University created the first rudimentary solid-state quantum processor [30]. The two-qubit superconducting chip is able to run elementary algorithms, such as a simple search, demonstrating quantum information processing with a solid-state device for the first time. Each of the two artificial qubits are made up of a billion aluminum atoms but they acted like a single one that could occupy two different energy states [30]. These states are akin to the "1" and "0", or "on" and "off" states of regular bits employed by digital computers. Moreover, qubits can be effectively place in a "superposition" of multiple states at the same time, allowing for greater information storage and processing power. These sorts of computations, though non-complex, have not been possible using solid-state qubits until now in part because scientists could not get the qubits to last long enough. While the first qubits of a decade ago were able to maintain specific quantum states for about a nanosecond, Schoelkopf and his team are able to maintain theirs for a microsecond, which is enough to run the simple algorithms. To perform their operations, the qubits communicate with one another using a "quantum bus" that uses photons to transmit information through wires connecting the qubits developed by the Yale group. The key dilemma was getting the qubits to switch "on" and "off" abruptly, so that they exchanged information quickly and only when the researchers wanted them to [30].

The team works to increase the amount of time the qubits maintain their quantum states so they can run more complex algorithms. They will also work to connect more qubits to the quantum bus. The processing power increases exponentially with each qubit added, so the potential for more advanced quantum computing is enormous [30].

VIII. Concluding remarks

To find optimal solutions, the quantum-based adaptive evolutionary algorithm AQMEA is proposed. It is a quantum technique for finding Pareto-optimal task allocation problem.

Although, there are several technological problems in building a quantum computer, quantum search provides a promising alternative for development some genetic algorithms and others multi-objective optimization techniques. However, it is some time before quantum computers are being used to solve complex problems.

Our future works will concern on a development the combination between quantum computing and evolutionary algorithms for finding Pareto-optimal solutions.

References

[1] Ameljańczyk, A.: *Multi-criteria optimization*, WAT Press, Warsaw (1986)

[2] Balicki J.: *An adaptive quantum-based multi-objective evolutionary algorithm for efficient task assignment in distributed systems*, Proc. of *The WSEAS Int. Conf. on Computers, July 22-26, 2009, Rodos Island, Greece*, WSEAS Press, pp. 417-422

[3] Balicki J.: *An adaptive quantum-based evolutionary algorithm for multiobjective optimization*, WSEAS Transactions on Systems and Control, Issue 12, Volume 4, December 2009, pp. 603-612

[4] Balicki J.: *Tabu programming for finding Pareto-optimal solutions*, WSEAS Transactions on Computers, *Vol. 7, 2008, pp. 2105-2114*

[5] Balicki J.: *Numerical experiments on Pareto-optimal task assignment representations by tabu-based evolutionary algorithm*, WSEAS Transactions on Information Science & Applications, *Vol. 5, 2008, pp.* 695-705

[6] Balicki J.: *Immune systems in multi-criterion evolutionary algorithm for task assignments in distributed computer system*. LNCS, 3528, pp. 51-56, Springer, Heidelberg, (2005)

[7] Benioff P.: *The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines*. Journal of Statistical Physics, vol. 22, pp. 563-591 (1980)

[8] Deutsch, D.: *Quantum computational networks*, in Proc. of the Royal Society of London A, vol. 425, pp. 73-90, (1989)

[9] Deutsch, D.: *Quantum Theory, the Church-Turing principle and the universal quantum* computer, in Proceedings of the Royal Society of London A, vol. 400, pp. 97-117 (1985)

[10] Dicarlo, L, *et al*.: Demonstration of two-qubit algorithms with a superconducting quantum processor. *Nature*, 460 (7252), pp. 240–4, (2009)

[11] DiVincenzo, D.: *The* Physical *Implementation of Quantum Computation*, IBM Report, 2000.

[12] Dorigo, M., Maniezzo, V., and Colorni, A., *The ant system: optimization by a colony of cooperating agents*. IEEE Transactions on Systems, Man, and Cybernetics, B 26(1), pp. 29–41, (1996)

[13] Feynman, R.: *Simulating physics with computers*, International Journal of Theoretical Physics, vol. 21, no. 6, pp. 467-488, (1982)

[14] Han K.–H. and Kim J.–H., *Genetic quantum algorithm and its application to combinatorial optimization problem*, in Proc. Congress on Evolutionary Computation, vol. 2, La Jolla, CA, July 2000, pp. 1354-1360, (2000)

[15] Han K.–H. and Kim J.–H.: *On setting the parameters of quantum-inspired evolutionary algorithm for practical applications.* in Proc. Congress on Evolutionary Computation, Canberra, Australia, pp. 178-184, (2003)

[16] Han K.–H. and Kim J.–H.: *Quantum-inspired evolutionary algorithm for a class of combinatorial optimization*, IEEE Trans. Evolutionary Computation, vol. 6, pp. 580-593, (2002)

[17] Han, K.–H., Park, K.–H., Lee, C.-H. and Kim, J.-H.: *Parallel quantum-inspired genetic algorithm for combinatorial optimization problem*, in Proc, Congress on Evolutionary Computation, vol.2, Seoul, Korea, May 2001, pp. 1422-1429 (2001)

[18] Hey, T. : *Quantum computing: An introduction*, in Computing & Control Engineering Journal. Piscataway, NJ: IEEE Press, June 1999, vol. 10, no. 3, pp. 105-112 (1999)

[19] Jang, J.-S., Han, K.-H. and Kim, J.-H.: *Quantum-inspired evolutionary algorithm-based face verification*, Lecture Notes in Computer Science. Berlin, Germany: Springer-Verlag, Proc. Genetic Evolutionary Computation Conf., pp. 2147-2156 (2003)

[20] Kim, J., Kim, J.-H., and Han, K.-H.: *Quantum-inspired Multiobjective Evolutionary Algorithm for Multiobjective 0/1 Knapsack Problems*, IEEE Congress on Evolutionary Computation Vancouver, Canada, July 16-21, pp. 9151-9156, (2006)

[21] Kim, K.-H., Hwang, J.-Y., Han, K.-H., Kim, J.-H. and Park, K.-H.: *A quantum-inspired evolutionary computing algorithm for disk allocation method*. IEICE Trans. Inform. Syst., E86-D, pp. 645-649 (2003)

[22] Korbicz, J.: *Artificial intelligence in technical diagnostics*. - Diagnostyka, 46, pp. 7-16, (2008)

[23] Plotkin, J. M.: *Hausdorff on ordered sets*. American Mathematical Society Bookstore, New York 2005.

[24] Shor, P.: *Algorithms for quantum computation: discrete logarithms and factoring*, in Proc. 35th Annual Symposium on Foundations of Computer Science, IEEE Press, November (1994)

[25] Talbi, H., Batouche, M., Draao, A.:. *A new quantum-inspired genetic algorithm for solving the Travelling Salesman Problem*, in Proc. IEEE International Conference on Industrial Technology, Vol. 3, December 2004, pp. 1192 - 1197, (2004)

[26] Talbi, H., Batouche, M., Draao, A.:. *A Quantum-Inspired Evolutionary Algorithm for Multiobjective Image Segmentation*, International Journal of Mathematical, Physical and Engineering Sciences, Vol. 1, No. 2, pp. 109-114, (2007)

[27] Vandersypen L. M. K., M. Steffen, G. Breyta, C. S. Yannoni, M. H. Sherwood and I. L. Chuang.: *Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance*. *Nature*, Vol. 414, pp. 883–887, (2001)

[28] Von Neumann J., Morgenstern T.: *The theory of games and economic behavior*, The Princeton University Press, Princeton 1944.

[29] Weglarz, J., Nabrzyski, J., Schopf, J.: *Grid resource management: State of the art and future trends.* Kluwer Academic Publishers, Boston (2003)

[30] -: *Scientists create first electronic quantum processor*, http://opac.yale.edu/news/article.aspx?id=6764, June 2009.

Jerzy M. Balicki is a university professor with the Faculty of Electronics, Telecommunications and Informatics, Gdansk University 11/12 Gabriela Narutowicza Street, 80-233 Gdańsk, Poland (e-mail: Balicki@eti.pg.gda.pl). He received the M.Sc. and Ph.D. degrees in Computer Science from Military University of Technology, Warsaw, Poland in 1982 and 1987, respectively. Then, he achieved habilitation D.Sc. from Technical University of Poznan in 2001. He was admitted as a university professor at Naval University of Gdynia in 2002. He is an author of three books and more than 120 scientific papers related to artificial intelligence, distributed computer systems, quantum algorithms and decision support systems.