GAHC: Improved GA with HC mutation

Radek Matousek and Lars Nolle

***Abstract* — This paper introduces a novel improved evolutionary algorithm, which combines genetic algorithms and hill climbing. Genetic Algorithms (GA) belong to a class of well established optimization meta-heuristics and their behavior are studied and analyzed in great detail. Various modifications were proposed by different researchers, for example modifications to the mutation operator. These modifications usually change the overall behavior of the algorithm. This paper presents a binary GA with a modified mutation operator, which is based on the well-known Hill Climbing Algorithm (HCA). The resulting algorithm, referred to as GAHC, also uses an elite tournament selection operator. This selection operator preserves the best individual from the GA population during the selection process while maintaining the positive characteristics of the standard tournament selection. This paper discusses the GAHC algorithm and compares its performance with standard GA.**

***Index Terms* — Elite tournament selection, HC mutation, Hill climbing Algorithm, Genetic Algorithm.**

I. Introduction

In mathematic, the term optimization refers to the study of problems in which one seeks to minimize or maximize an objective function *f* by systematically choosing the values of the independent real or integer variables from within a given set **X**. The variables are the parameters **x** to be optimized for a given optimization task. The goal of an optimization algorithm is to find $\mathbf{x}_{\text{opt}}$ which fulfills equation 1:

$$\mathbf{x}_{\text{opt}} = \arg\operatorname*{opt}_{\mathbf{x}\in\mathbf{X}} f(\mathbf{x}), \tag{1}$$

where argopt returns the value (or values) of **x** in the set **X** that minimize (or maximize) the objective function *f*.

If the objective function is convex over the region of interest, any local extreme value of the function will also be a global extreme. For this class of problems, fast and robust numerical optimization techniques exist. However, if the objective function is nonlinear and/or multi-modal, deriving at the global solution can be difficult. In such a case, soft computing methods can offer a solution.

Manuscript received July 21, 2007. This work was supported by the Czech Ministry of Education in the frame of MSM 0021630529 Research Intention *Inteligent Systems in Automation*.

R. Matousek is with the Brno University of Technology, Faculty of Mechanical Engineering, Institute of Computer Science and Automation, Technicka 2896/2, 61600 Brno, Czech Republic (corresponding author to provide e-mail: matousek@fme.vutbr.cz).

L. Nolle is with the Nottingham Trent University, School of Computing and Informatics, Clifton Lane, NG11 8NS, UK; (e-mail: lars.nolle@ntu.ac.uk).

Examples of these methods are *Genetic Algorithm* (GA) and *Hill-climbing Algorithm* (HCA). A common feature of GA and HCA is the *binary representation* of the vector **x**. In general [1], GA is a global search method, whereas HCA is a local optimization method [6]. By combining a local search technique, in this case HCA, with a global search method, i.e. GA, the resulting algorithm could be more powerful than the individual algorithms. In this case, HCA is incorporated into GA by using it as a mutation operator. Also, an innovated version of tournament selection denoted as an elite tournament is used in this work. Further, a proper design of the HCA can increase an optimization property of the HCA.

II. Parameter Encoding

A. Binary representation of the task

The coding representation of a problem is based on the binary vectors of length *n*.

$$\mathbf{a} = \{a_1, a_2, \ldots, a_n\} \in \{0,1\}^n \tag{2}$$

In the context of genetic algorithms, these vectors called individuals. A set of individuals is denoted as a population **P**.

$$\mathbf{P} = \left(a_{i,j}\right)_{\substack{i\in\{1,2,\ldots l\}\\ j\in\{1,2,\ldots n\}}} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{l,1} & a_{l,2} & \cdots & a_{l,n} \end{pmatrix} \tag{3}$$

The real or integer parameters **x** are derived from the binary vector (2), respective (3), by means of a specific transformation Γ (equation 4).

$$\mathbf{x} = \Gamma(\mathbf{a}) \tag{4}$$

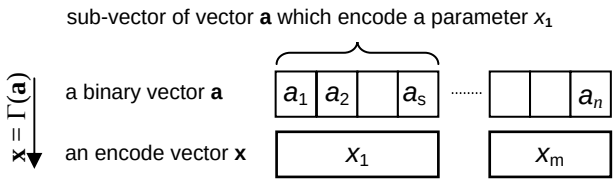

Fig. 1: Parameters encoding scheme.

B. Hamming metric

A distance function ρ is a function which is used as a metric that defines a distance between elements of a set and fulfills the ground axioms of the metric (positive definiteness, symmetry, triangle inequality). A set with a metric is called a metric space, which is a binary metric space in case of binary representation. The *Hamming distance* ρ_H between two binary vectors of equal length is the number of positions for which the corresponding symbols are different. Let **a**, **b** are binary vectors of length *n* and *a*, *b* its elements, then the Hamming distance can be calculated as follows:

$$\rho_H(\mathbf{a},\mathbf{b}) = \sum_{i=1}^{n} |a_i - b_i| . \quad (5)$$

C. Domain of definition

The domain of interval *D* of the encoded real or integer vectors **x** (4) is given by (6) and it represent the feasible area of the possible solution(1).

$$D = \prod_{i=1}^{m} [r_i, s_i] = [r_1, s_1] \times [r_2, s_2] \times \ldots \times [r_m, s_m] \quad (6)$$

In case of binary representation, the *D* area can be approximated using an orthogonal grid (Fig. 2).

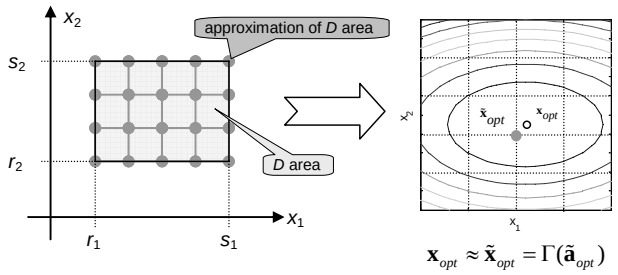

Fig. 2: The solution space sampling and its effect

Each individual in the population **P** is evaluated using an objective function *f* on the domain definition *D*.

$$f : D \to \Re \quad (7)$$

III. Implemented HCA

Hill climbing is an optimization technique which belongs to the class of local search methods [2, 4]. The algorithm is started with an initial solution to the problem, and subsequently makes small changes to the solution, each time improving it a little bit. At some point the algorithm arrives at a point where it cannot find any improvement anymore, which is when the algorithm terminates. Ideally, at that point the solution found should be close to optimal solution, but it is not guaranteed that hill climbing will ever come close to this optimal solution.

A. Basic concept

The most important step of HCA is to generate a close neighborhood of an initial (previous) solution. This solution can be denoted as a kernel of the HC transformation. The next solution, i.e. kernel, is chosen as the best from the set of neighborhood solutions.

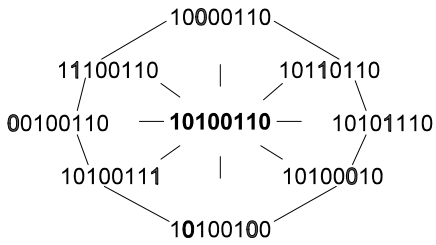

Fig. 3: An example of neighborhood generating.

The Hamming metric is used for measuring the distances between the previous solution and the newly generated solution from the neighborhood. All solutions must belong in discrete area *D*. The area $D \in \Re$ is given by the relation

$$\Gamma : \{0,1\}^n \to D . \quad (8)$$

Because $\mathbf{x} = \Gamma(\mathbf{a})$ we can find the optimal solution of a task as

$$\mathbf{a}_{opt} = \arg \min_{\mathbf{a} \in \{0,1\}^l} f\left(\Gamma(\mathbf{a})\right) \quad (9)$$

Below the neighborhood is defined, which makes it possible for every acceptable kernel $\mathbf{a}_{\text{kernel}}$ to determine its neighborhood $\mathbf{a}_{\text{neighbor}} \in S(\mathbf{a}_{\text{kernel}})$. The choice of the transfer function *S* will determine the behavior of HCA.

An example of HCA pseudo-code is following

% binary HCA (minimization)
$\mathbf{a}_{opt}$ *= the first iteration (random generated vector)*
repeat
  $\mathbf{a}_{kernel} = \mathbf{a}_{opt}$
  $\forall \mathbf{a}_{neighbor} = S(\mathbf{a}_{kernel})$
  $\mathbf{a}_{\text{opt}} = \arg \min_{\mathbf{a} \in S(\mathbf{a}_{\text{kernel}})} f\left(\Gamma(\mathbf{a})\right)$
until $f\left(\Gamma(\mathbf{a}_{\text{opt}})\right) \geq f\left(\Gamma(\mathbf{a}_{\text{kernel}})\right)$

Fig. 4: Pseudo-code of the Hill-climbing algorithm.

B. Transfer function

The HCA used in this work is based on an arbitrary but fixed given set of transformations of binary vectors.

$$\mathbf{a} \in \{0,1\}^n, \quad n \in \mathbb{N} \quad (n\text{...length of a binary string}) \quad (10)$$

Let *H* be a set of designed transformations and *t* be a given transformation.

$$H = \{t_0, t_1, ..., t_n\} \quad (11)$$

For the mapping of a binary vector $\mathbf{a}_{\text{kernel}}$ on the $\mathbf{a}_{\text{neighbor}} \in A_{\text{neighbor}}$ the a transformation $t \in H$ is used, as is shown in the following equations

$$A_{neighbour} = \{\mathbf{a}_1, \ldots, \mathbf{a}_c\}, \quad \mathbf{A}_{neighbour} = \begin{pmatrix} \mathbf{a}_1 \\ \vdots \\ \mathbf{a}_c \end{pmatrix}, \quad \text{pro } c \in \mathbb{N} \quad (12)$$

$$t : \mathbf{a}_{\text{kernel}} \to A_{neighbour}, \quad \text{i.e. } t : \{0,1\}^n \to \left(\{0,1\}^n\right)^c, \quad (13)$$

where c is the cardinality of the set A_{neighbor}. The cardinality depends on the chosen transformation t and the length n of the binary vector $\mathbf{a}_{\text{kernel}}$.

$$c_k\left(t_k, n\right) = \left|A_{neighbour}\right| = \binom{n}{k}, \quad \text{for } k \in \{0,1,\ldots,n\}, \quad (14)$$

where index k denotes the relation to the concrete element from the set H according (11).

For the realization of the set of transformation H, a set of matrices $\mathbf{M}$ is introduced. For every given transformation t_k there exists a corresponding matrix $\mathbf{M}$, which is of the order k and is denoted by $\mathbf{M}_k$. Please note that in this case the order has a different meaning from the standard square matrix order.

$$t_k, n \Leftrightarrow \mathbf{M}_k, \quad \text{for } k \in \{0,1,\ldots,n\} \quad (15)$$

Definition: The matrix $\mathbf{M}$ of order k, shortly $\mathbf{M}_k$, is a matrix where the rows represent all the points of the Hamming spaces $\mathbb{H}^n$ with the distances k from the origin (i.e. the zero vectors of length n) in the sense of metrics measurement ρ_{H}. □

Possible examples of matrix $\mathbf{M}$ are outlined in the following equations (16).

$$\mathbf{M}_0 = \begin{pmatrix} 0_{1,1} & 0_{1,2} & \cdots & 0_{1,n} \end{pmatrix}$$

$$\mathbf{M}_1 = \begin{pmatrix} 1_{1,1} & 0_{1,2} & \cdots & 0_{1,n} \\ 0_{2,1} & 1_{2,2} & & \\ \vdots & & \ddots & \\ 0_{c_1,1} & & & 1_{c_1,n} \end{pmatrix}$$

$$\mathbf{M}_2 = \begin{pmatrix} 1_{1,1} & 1_{1,2} & 0_{1,3} & \cdots & 0_{1,n} \\ 1_{2,1} & 0_{2,2} & 1_{2,3} & & 0_{2,n} \\ \vdots & & & & \\ & & & \ddots & \\ 0_{c_2,1} & & & 1_{c_2,n-1} & 1_{c_2,n} \end{pmatrix}$$

$$\vdots$$

$$\mathbf{M}_{n-1} = \begin{pmatrix} 0_{1,1} & 1_{1,2} & \cdots & 1_{1,n} \\ 1_{2,1} & 0_{2,2} & & \\ \vdots & & \ddots & \\ 1_{c_{n-1},1} & & & 0_{c_{n-1},n} \end{pmatrix}$$

$$\mathbf{M}_n = \begin{pmatrix} 1_{1,1} & 1_{1,2} & \cdots & 1_{1,n} \end{pmatrix} \quad (16)$$

For the calculation of the $\mathbf{A}_{neghbour}$ matrix, it is necessary to introduce the operation for the so-called vector replication of a binary vector $\mathbf{a}_{kernel}$. An operation matrix $\mathbf{A}_{kernel}$ is formed and contains identical copies of the binary vector $\mathbf{a}_{jadro}$ in rows. The number of rows in the matrix $\mathbf{A}_{kernel}$ is equal to the number of rows in the given matrix $\mathbf{M}$.

$$\mathbf{A}_{kernel} = \begin{pmatrix} \mathbf{a}_{1,kernel} \\ \vdots \\ \mathbf{a}_{c,kernel} \end{pmatrix} \quad (17)$$

Applying the principle of addition in modular arithmetic (mod 2) to (14), respectively (15), leads to the following transformation t (13)

$$t_k : \ \mathbf{A}_{neighbor} = \mathbf{A}_{kernel} \oplus \mathbf{M}_k, \quad for\ t_k \in H\ a\ k = 0,1,...,n \ (18)$$

The transformation t_k generates the complete set of vectors, which have the distance k from the origin in terms of the metrics ρ_H.

$$t_k \Rightarrow \rho_H(\mathbf{a}_{kernel}, \mathbf{a}_{neighbor}) = k, \quad for\ \forall \mathbf{a}_{neighbor} \in A_{neighbor} \quad (19)$$

The generalization of equation (18) for a arbitrary but fixed choice of the elements from the set of $\mathbf{H}$ is evident. The set of chosen, and for HCA fixed, transformation is denoted as $\mathbf{H}_v$.

$$H_v \subseteq H \quad (20)$$

The set $\mathbf{H}_v$ definitely determines the transformation S, which is linked with transformation from this set.

$$H_v \Leftrightarrow S, \quad thus \quad S : \mathbf{A}_{NEIGHBOR} = \begin{pmatrix} \mathbf{A}_{kernel,k_1} \oplus \mathbf{M}_{k_1} \\ \vdots \\ \mathbf{A}_{kernel,k_v} \oplus \mathbf{M}_{k_v} \end{pmatrix}, \quad (21)$$

Where $k_i \in I$ and I is the index set of the chosen elements from $\mathbf{H}$, which is defined by the choice $\mathbf{H}_v$. The cardinality $c_{NEIGHBOR}$ of the set $A_{NEIGHBOR}$, which is arrived at by means of the set of transformations H_v, is determined by sum of all partial cardinalities in equation (14), i.e.

$$A_{NEIGHBOR} = \bigcup_{k_i} A_{neighbor,k_i} \quad (22)$$

$$c_{NEIGHBOR} = \sum_{k_i \in I} \binom{n}{k_i} \quad (23)$$

Remark 1: The equation for the sum of binomial coefficients is obtained for the limited case. This equation states that the whole space of $n + 1$ transformations by (11) represents a full search of the Hamming metrics space.

$$\sum_{i=0}^{n} \binom{n}{i} = 2^n \quad (24)$$

Remark 2: One consequence of the previous remark is rather theoretic; algorithms that carry out an exhaustive space searching are guaranteed to find the global solution in a given universe, but at the price of uncontrollable combinatorial expansion.

$$\mathbf{a}_{\substack{global \\ optimum}} = \left[\arg \min_{a \in \{0,1\}^n} f\left(\Gamma(\mathbf{a})\right)\right], \quad \text{pro } \left|A_{NEIGHBOR}\right| \to 2^n \ (25)$$

C. Implementation of the proposed Hill Climbing Algorithm

For the practical implementation of the mutation operator in GAHC a variant of HCA denoted as HC12 (conjunction of the HC1 and HC2 algorithms) was used [4]. This HCA uses the set of transformations $H_v = \{t_0, t_1, t_2\}$.

name	HC1	HC2	HC12
set of transformations	t_0, t_1	t_0, t_2	t_0, t_1, t_2
M matrices	$\mathbf{M}_0$, $\mathbf{M}_1$	$\mathbf{M}_0$, $\mathbf{M}_2$	$\mathbf{M}_0$, $\mathbf{M}_1$, $\mathbf{M}_2$
cardinality of neighborhood	$1+n$	$1+\binom{n}{2}$	$1+n+\binom{n}{2}$
$\rho_H(a_{kernel}, \forall a_{neighbour})$	1	2	1 and 2

The designed variants of HCA prevent a decrease of the objective function value because the t_0 transformation is always contained. Examples of $\mathbf{A}_{neighbor}$ vector coverage of the solution space can bee seen in the figure below.

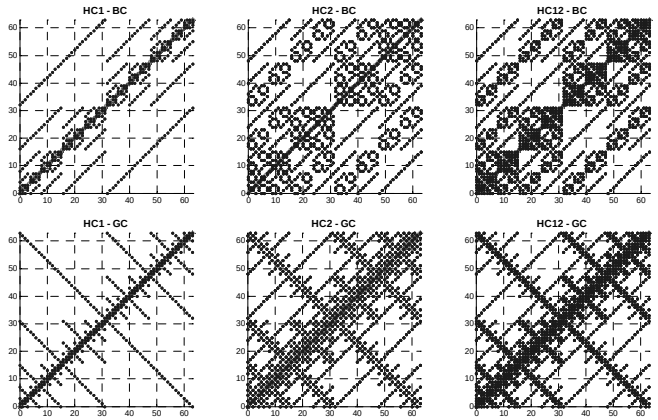

Fig. 5: Examples of coverage of a solution space for different transformations H_v and selected encoding methods, where BC denotes a direct binary encoding, whereas GC represents a Gray binary encoding.

Examples of stand-alone HCA search runs for the well-known Rosenbrock's test function, also known as F2 or banana valley function, are shown in figure 6 (standard variant HC1) and in figure 7 (advanced variant HC12).

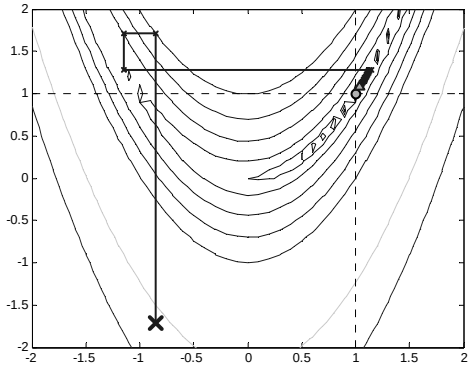

Fig. 6: Example of a search run using the stand-alone HCA variant HC1.

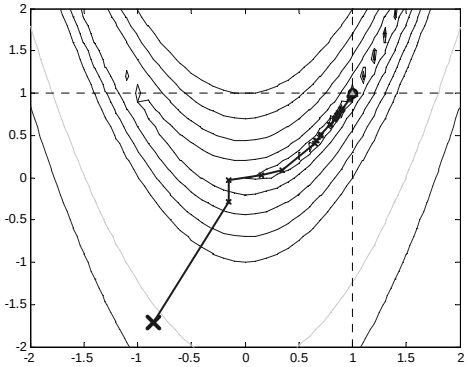

Fig. 7: Example of a search run using the stand-alone HCA variant HC12.

In these examples, each parameter was encoded using 15-bit Gray code representation. The standard HC1 variant needed 1527 iteration steps and found the solution $F2_{min}(x_1=1.0375, x_2=1.0764) = 0.0014$. The advanced HC12 variant needed, starting at the same initial conditions, only 46 iteration steps and found a better solution $F2_{min}(x_1=0.9999, x_2=0.9998) = 0.0000$.

IV. Elite Tournament Selection.

Tournament selection is one of many methods for selecting offspring in genetic algorithms. It runs a "tournament" among a number of randomly chosen individuals from the population and selects the winner, i.e. the one with the best fitness, for crossover. This chapter introduces briefly an enhanced version of tournament selection, called *elite tournament* [4, 5]. This improved selection method overcomes the main problem of the standard tournament selection, which is that it does not guarantee the reproduction of the best solution contained in the current generation.

A. Background

The selection operator is intended to improve the average quality of the population **P** (3) by giving individuals of higher quality a higher probability to be copied into the next generation, i.e. selection operator s: $\mathbf{P}_{BS} \rightarrow \mathbf{P}_{AS}$. Here, $\mathbf{P}_{BS}$ refers to the population before selection and $\mathbf{P}_{AS}$ refers to the population after selection. In other words, the selection operator focuses the search on promising regions in search space.

The tournament selection operator is very popular because of its properties [GD91]. It is probably the best selection method in many respects. However, one shortcoming of this selection mechanism is that it does not guarantee the survival of the best individual through the selection process s: $\mathbf{P}_{BS} \rightarrow \mathbf{P}_{AS}$. In the case of elite tournament selection, this disadvantage of the classical tournament selection operator is eliminated, while preserving all the advantages of the classical tournament selection.

Tournament selection works as follows: Choose randomly t individuals from the population $\mathbf{P}_{BS}$ and copy the best

individual from this tournament group into the population $\mathbf{P}_{AS}$. This step is repeated l times. The probability of selecting p_s of the best individual is

$$p_s(t,l) = 1 - \left(\frac{t-1}{l}\right)^{t \cdot l}. \quad (26)$$

Elite tournament selection is the same as tournament selection; it copies l times the best individual from a group to population $\mathbf{P}_{AS}$, but the tournament groups are formed differently: the first individual in group i, where $i \in (1, 2, \ldots, l)$, is the ith individual from population $\mathbf{P}_{BS}$, the rest of group is selected randomly. The time complexity of this algorithm is the same as tournament selection, i.e. $\mathcal{O}(n)$. The probability of selection of the best individual is

$$p_s(t,l) = 1 \quad (27)$$

B. Evaluation and analysis

For the analysis of the proposed algorithms the *takeover time* test was used. In the experiments a population size n=100 was used. The following empirical results (Fig. 8) were obtained from 1000 independent runs during selection processes only.

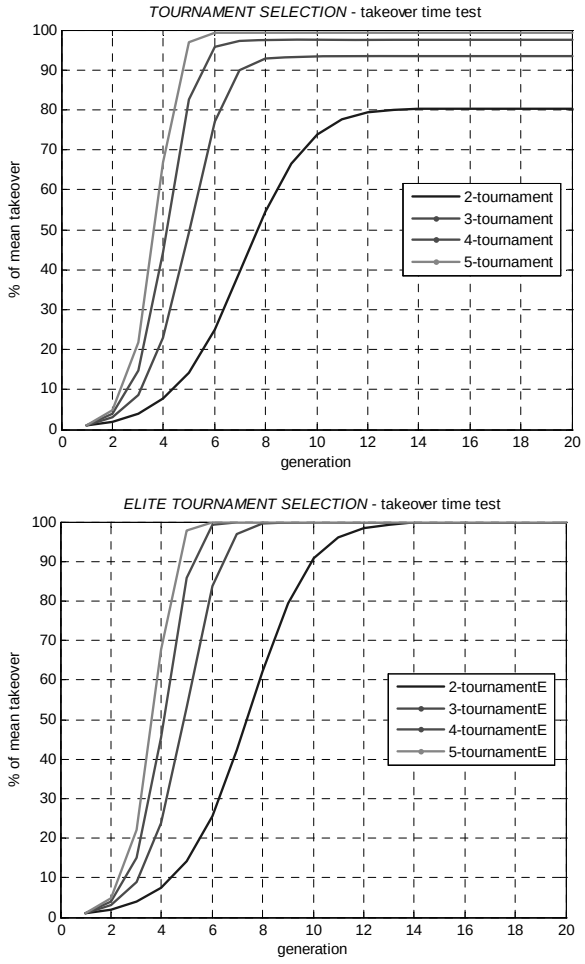

Fig. 8: Average takeover times over 1000 runs with tournament selection (tournament) and elite tournament selection (tournamentE) for different tournament sizes t={2,3,5,5} using populations of 100 individuals. The graphs represent the proportion of the population consisting of the best individual as a function of generation.

V. GAHC

Exploration versus exploitation is a well-known issue in Evolutionary Algorithms. An unbalanced search may waste many iteration steps, i.e. computational expensive fitness evaluations, or lead to premature convergence. A very successful strategy is to combining global and local search methods to improve global optimization capabilities. However, HCA, in the version HC12, is strictly speaking not a local search technique.

A. Background

The following primary requirements for GAHC design were identified:

- The global search capability of the GA must be preserved or improved. Premature convergence of the GA should not occur.
- The local search feature of the GA should only require a few iteration steps, i.e. fitness evaluations, during the search for the optimum.
- The computation time for solving the optimization task should be less than for the stand-alone GA.

GAHC, an improve Genetic Algorithm, addresses this requirements by using an intelligent mutation technique, which can replace or supplement the standard GA mutation operator.

In this work, HCA is used as mutation operator:

1. chose an individual $\mathbf{I}_i$,
2. determining the kernel position,
3. applying of the HCA on the kernel, using the entire individual for objective function evaluation.

The individual, kernel position and kernel size can be chosen randomly or based on some heuristic, that could exploit some existing inside into the optimization problem at hand. The number of HCA kernels and their sizes are determined by the standard GA mutation operator. For example, if 10 HCA kernels of length 5 bits would be used, this would result into a standard bit mutation of 50bits. The HCA kernel is chosen to contain 10 bits for the GAHC in order to reduce computation time and to improving the global search behavior.

The size of the HCA kernel is depended on the bit length of an individual. The basic principle of the application of the HCA mutation operators can be seen from the figure (Fig. 9).

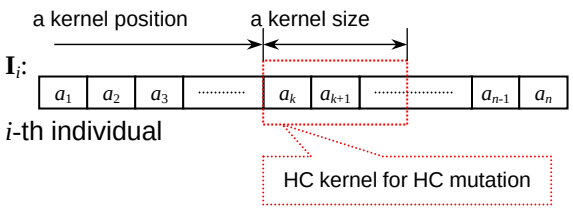

Fig. 9: The principle application of HCA as mutation operator for GAHC.

B. Evaluation and analysis

In order to evaluate the new GAHC algorithm, Ratrigin's function F6 (equation 28) was chosen for the experiments.

$$F_6(\mathbf{x}) = 10n + \sum_{i=1}^{n}\left(x_i^2 - 10\cos(2\pi x_i)\right) \qquad (28)$$

$$-5.12 \le x_i \le -5.12,\ \min F_6(\mathbf{x}) = F_6(0,...,0) = 0$$

Rastrigin's function F6 is based on the power function with the addition of cosine modulation to introduce many local minima. Thus, the test function is highly multimodal. However, the locations of the minima are equally distributed.

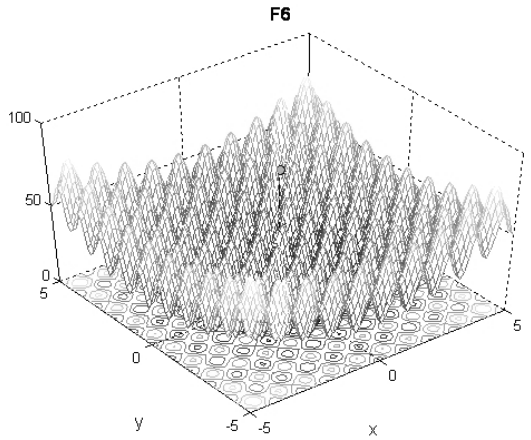

Fig. 10: Visualization of Rastrigin's function F6 in the range from -5 to 5. The global optimum is positioned at point [0, 0].

Results of the test runs [5] for the GA and GAHC variant are presents in figure 11. The number of runs which were carried out in each tests was 1000. The population size was chosen to be 50 individual and each individual had a length of 50 bits.

The GA parameters chosen were the followings:

parameter	value
GA test suite	F6, 5 optimized variables
GA selection	Elite tournament selection, 3
GA crossover	80% of individual in 3cat
GA mutation	mutation probability 0.02
GAHC	10 HCA kernel of size 5 bits

VI. Conclusion

This paper introduced GAHC, a novel improved evolutionary algorithm, which combines genetic algorithms and hill climbing.

The main principles of the Hill-Climbing algorithm were explained and a powerful variant, referred to as HC12, was introduced. These algorithms can be used as stand-alone optimization algorithms or they can be used as an effective and efficient mutation algorithm within a Genetic Algorithm.

Experiments, using the well-known test functions F6 for GAHC and F2 for the HCA and HC12 variants, have shown that if the Hill-Climbing method is used as a mutation operator, it improves the local search capabilities of Genetic Algorithms while preserving their global search capabilities.

Finally, was also demonstrated that a new tournament selection operator, referred to as elite tournament selection, further improved the performance of the new hybrid algorithm.

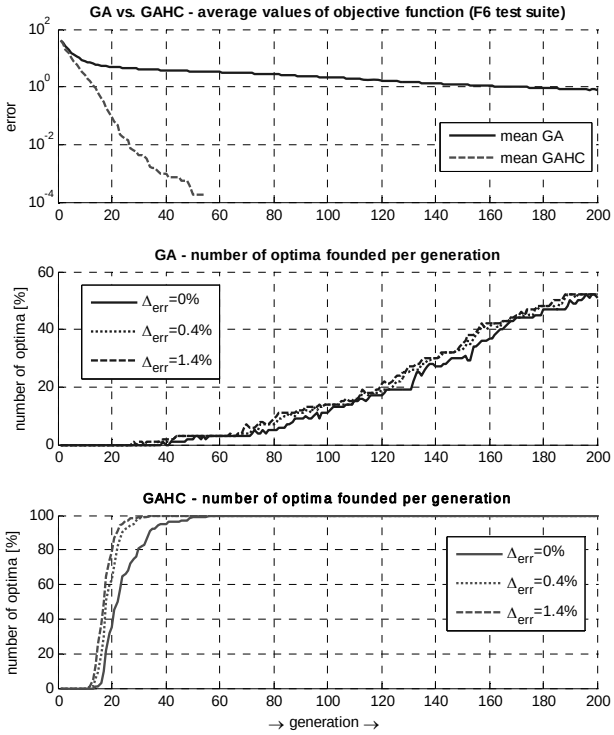

Fig. 11: A visualization of the GA and GAHC test runs. The second and third graphs represent the number of optima founded per generation for 1000runs.

Acknowledgment

This work was supported by the Czech Ministry of Education in the frame of MSM 0021630529 *Intelligent Systems in Automation*.

References

[1] David E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, 1989

[2] Halim C., *Developing Combined Genetic Algorithm—Hill-Climbing Optimization Method for Area Traffic Control*, J. Transp. Engrg., Volume 132, Issue 8, pp. 663-671, August 2006

[3] Matousek, R., Bednar, J., *Elite Tournament Selection*, in the P. Osmera Proceeddings of Mendel 2005 Soft Computing conference, Brno, Czech Republic, 2005

[4] Matousek, R., *GA (GA with HC mutation) – implementation and application (in Czech)*, master thesis, Brno University of Technology, Brno, Czech Republic, 1995

[5] Matousek, R., *Selected Methods of Artificial Intelligence – Implementation and Application (in Czech)*, Ph.D. thesis, Brno, Brno University of Technology, Czech Republic, 2004

[6] Mitchell, M., Holland, J. H., *When Will a Genetic Algorithm Outperform Hill Climbing?* In J. D. Cowan, G. Tesauro, and J. Alspector (editors), Advances in Neural Information Processing Systems 6. San Mateo, CA: Morgan Kaufmann, 1994.