Applied Soft Computing 12 (2012) 2924–2932

Contents lists available at SciVerse ScienceDirect

Applied Soft Computing

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

Crossover-based local search in cooperative co-evolutionary feedforward neural networks

Rohitash Chandra*, Marcus Frean, Mengjie Zhang

School of Engineering and Computer Science, Victoria University of Wellington, Wellington, New Zealand

A R T I C L E I N F O

Article history:
Received 2 March 2011
Received in revised form 24 March 2012
Accepted 15 April 2012
Available online 4 May 2012

Keywords:
Memetic algorithm
Local search
Cooperative coevolution
Feedforward networks

A B S T R A C T

Cooperative coevolution has been a major approach to neuro-evolution. Memetic algorithms employ local search to selected individuals in a population. This paper presents a new cooperative coevolution framework that incorporates crossover-based local search. The proposed approach effectively makes use of local search without adding to the computational cost in the sub-populations of cooperative coevolution. The relationship between the intensity of, and interval between the local search is empirically investigated and a heuristic for the adaptation of the local search intensity during evolution is presented. The method is used for training feedforward neural networks on eight pattern classification problems. The results show an improved performance in terms of optimisation time, scalability and robustness for most of these problems.

1. Introduction

Memetic algorithms (MA) [15] typically combine population-based evolutionary algorithms with local refinement which provides intensification. The meme is considered to be an individual which goes through local refinement in memetic algorithms. Much attention has been given to finding efficient local search techniques for memetic algorithms and it has been shown that evolutionary algorithms (EAs) can be used as effective local search techniques [12,13]. In crossover-based local search [13,14], efficient crossover operators which have local search properties are used for local refinement with a population of a few individuals. Promising results have been shown in comparison with other evolutionary algorithms for global optimisation problems [14].

Cooperative coevolution (CC) decomposes a large problem into smaller subcomponents and solves them independently [19]. The subcomponents are represented using sub-populations which are genetically isolated and the cooperation takes place for fitness evaluation. Cooperative co-evolution has been shown to be effective for neuro-evolution of feedforward and recurrent networks using a variety encoding schemes for problem decomposition [8,7,2,3].

There has been much research in using local refinement with standard evolutionary algorithms which consider a single population. The success of local search in conventional evolutionary algorithms gives the motivation for using local search in cooperative coevolution. A major focus of memetic algorithms is on using efficient local search methods in order to balance between global and local search during evolution. It is important to investigate how often (*local search interval*) and for how long (*local search intensity*) to apply them.

This work presents a new memetic cooperative coevolution framework which utilises the strength of local refinement in training feedforward neural networks. We employ crossover-based local search and name the proposed method *crossover-based local search in cooperative coevolution* (XLCC). The relationship between the local search interval and intensity is empirically investigated and a heuristic for adapting the local search intensity during evolution is proposed. XLCC is used to train feedforward networks on pattern classification problems. The goal of this paper is to develop a memetic cooperative co-evolutionary framework which reduces the overall training time and provides a better guarantee for convergence. The performance of XLCC is compared with standard cooperative coevolution.

The main contribution of the paper is in the incorporation of crossover-based local search in cooperative coevolution, and the adaptation of local search intensity during evolution. The proposed framework is specifically targeted at training a fixed neural network topology in which only the weights of the neural network are evolved.

The rest of the paper is organised as follows. Section 2 presents the background on memetic algorithms with an emphasis on using evolutionary algorithms for local search. The cooperative coevolution framework and problem decomposition methods for neural networks are introduced. Section 3 presents the memetic cooperative coevolution framework which employs crossover-based local

* Corresponding author.
E-mail address: c.rohitash@gmail.com (R. Chandra).

1568-4946/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.asoc.2012.04.010

search. Section 4 presents experimental results and Section 5 concludes the paper with a discussion on future work.

2. Background

2.1. Memetic algorithms

Memetic algorithms [15] typically combine population-based evolutionary algorithms with local search (LS) in order to provide an improved global solution. The local search is also known as local refinement or intensification. Memetic algorithms also include the combination of EAs with problem dependent heuristics, approximate methods and special recombination operators [17].

Moscato introduced memetic algorithms using a genetic algorithm for diversification with local search for intensification [16]. A growing field of interest is in using evolutionary algorithms for local search methods in memetic algorithms. The EA for local search has a small population size which is evolved for a short duration [12,13,20,14].

Kazarlis et al. [12] introduced the concept of micro genetic algorithms for local search where a population of few individuals was employed as a generalised hill-climber intended for intensification and a genetic algorithm with a larger population was used for diversification. Lozano et al. [13] presented a real-coded memetic algorithm with crossover hill-climbing. This algorithm maintains a pair of parents, consisting of the solution being refined and the best solution. The crossover operation is performed on the pair until some number of offspring has been generated. The method performed better than previous memetic algorithms presented in the literature.

Molina et al. [14] used evolution strategies [9] with a small population as a subordinate EA for local search. They used a steady-state genetic algorithm [10] as the master EA which has the property of high population diversity. The method showed good results for continuous problems with high dimensionality when compared with its counterparts from the literature.

2.2. Cooperative coevolution

Cooperative coevolution (CC) is an evolutionary computation framework inspired from nature which divides a large problem into subcomponents [19]. The subcomponents are implemented as sub-populations.

Problem decomposition is how a problem is broken into subcomponents. The appropriate problem decomposition method depends entirely on the nature of the optimisation problem. The original cooperative coevolution framework decomposed the problem in a way such that a single sub-population is used for each variable [19]. The algorithm begins by initialising and cooperatively evaluating each of the individuals of the respective sub-populations. All the sub-populations are evolved in a round-robin fashion for the *depth* of *n* generations. This is also known as a *cycle*.

2.3. Problem decomposition for feedforward networks

In the case of evolving neural networks using cooperative coevolution, problem decomposition is also referred to as the encoding scheme. The major encoding schemes include those based on the *neuron level* and *synapse level*. The neuron level encoding scheme employs each neuron in the hidden layer as the main reference point for the respective subcomponent. Each subcomponent consists of the incoming and outgoing connections. The cooperative coevolution model for evolving artificial neural networks [6] and multi-objective cooperative networks [5] build subcomponents by mapping all input and output connections from the respective hidden neuron. They have been used for training feedforward network architectures. The synapse level encoding scheme employs a single subcomponents for each weight or connection in the neural network. This breaks the network to its lowest level of granularity [7].

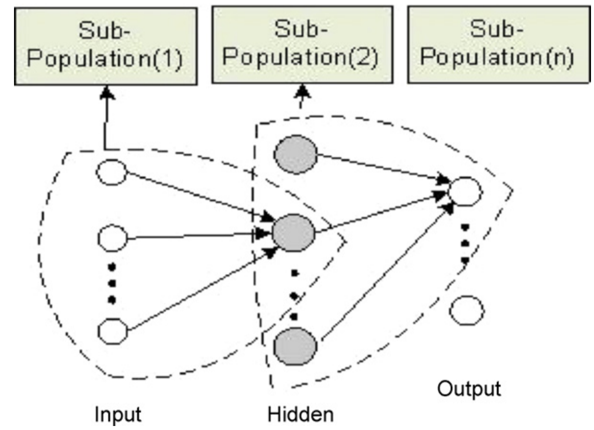


Fig. 1. The NSP encoding scheme. The same strategy is used in the rest of the neurons in the hidden and output layer [2]. This decomposition is also possible for a neural network with more than one hidden layer.

The authors have presented an encoding scheme called neuron-based sub-population (NSP) which showed better performance than the neural and synapse encoding schemes from the literature for pattern classification problems [2]. In this work, the NSP is used for training feedforward networks in which each sub-population in a layer composed of the following.

1. Hidden layer sub-populations: weight-links from each neuron in the *hidden* layer connected to all *input* neurons and the bias of the *hidden* neuron.
2. Output layer sub-populations: weight-links from each neuron in the *output* layer connected to all *hidden* neurons and the bias of the *output* neuron.

Each neuron in the hidden and output layer acts as a reference point to its sub-populations as shown in Fig. 1. Note that this decomposition is also possible for a neural network with more than one hidden layer.

3. The proposed memetic cooperative coevolution framework

Evolutionary algorithms are composed of a population of individual solutions. In the case of neuro-evolution, the individual solutions represent the weights and biases of the neural network. Memetic algorithms have mainly been developed using evolutionary algorithms which have a single population of individuals. A memetic cooperative coevolution computation framework has to consider the computational costs of employing local search for each sub-population. The respective individuals in a sub-population that undergo individual search only represent a subset of the large problem. In order to apply local search, an individual has to be concatenated with the best individuals in the rest of the sub-populations. Therefore, given *n* sub-populations, *n* local searches would be required, adding to the computational cost. We propose a framework which takes advantage of local search while at the same time lowering the computational cost of having a separate search for every sub-population. Rather than employing a search for each sub-population, our proposed framework employs local search only when the *cycle* is complete. The completion of a cycle in cooperative coevolution indicates that all the respective sub-populations have been evolved for a given number of generations.

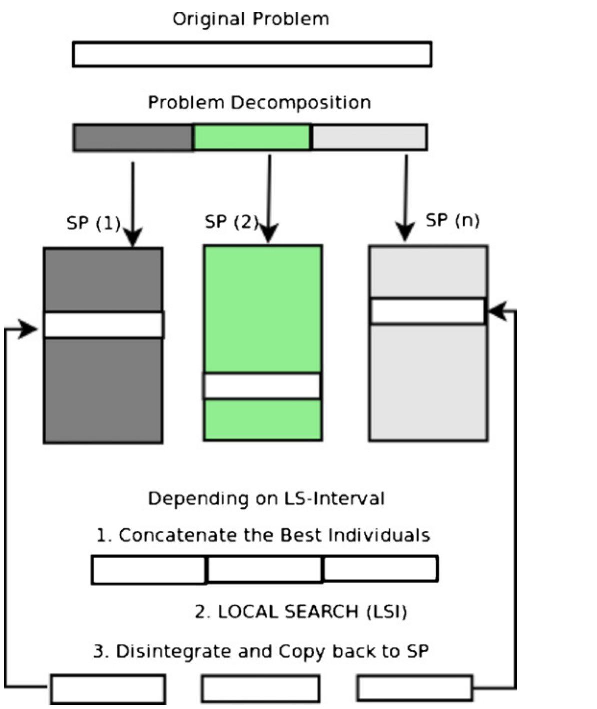


Fig. 2. Proposed XLCC framework to employ LOCAL SEARCH after concatenating the best individuals from each sub-population (SP) at the end of each cycle.

The *local search interval* (LS-Interval) determines how often to employ local search. For instance, the LS-Interval of 1 indicates that local search should be employed after every cycle of cooperative coevolution. The local search is applied according to the *local search intensity* (LSI), which is expressed in terms of k generations.

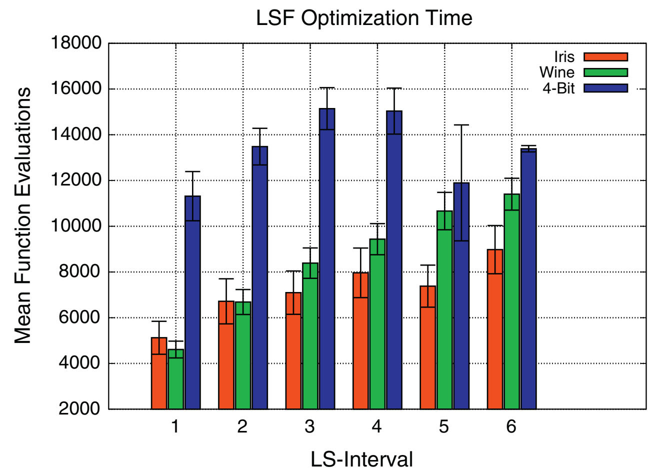
Algorithm 1 *(Memetic cooperative coevolution framework).*

– Encode the neural network using an appropriate encoding scheme
– Randomly initialise all sub-populations
– Cooperatively evaluate each sub-population
while NOT termination **do**
  for LS-Interval **do**
    for each sub-population **do**
      for depth of n generations **do**
        Create new individuals using genetic operators
      end for
    end for
  end for
    – Concatenate the best individuals from each sub-population into meme M
    – Local search on M according to the LSI
  (i) Decompose M according to the respective sub-populations
  (ii) Replace the worst individuals of the respective sub-populations with decomposed M
end while

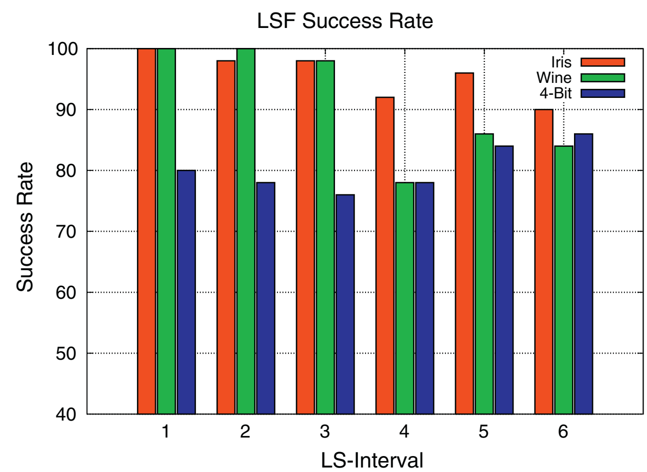
The details of the proposed memetic cooperative coevolution framework are given in Algorithm 1. The algorithm assumes that it has been given the best parameters for the evolutionary algorithm such as its population size, crossover and mutation rate. The algorithm begins by encoding the neural network into the sub-population according to the respective cooperative coevolution encoding scheme (either synapse or neuron level [7,2]). The specific encoding scheme for this work is NSP which has been discussed earlier.

The crossover-based local search employs a population of a few individuals which is also referred as the local search population. The goal of the cooperative coevolution population is to provide diversity, and the local search population provides intensification. All the individuals of the respective sub-population are initialised with random real values. Each individual chromosome is then concatenated with the best individuals of the rest of the sub-populations and then encoded into a neural network and evaluated as done in [18].

The algorithm proceeds as a standard evolutionary algorithm which employs genetic operators such as selection, crossover and mutation to create new offspring for all the sub-populations. Each sub-population is evolved for a depth of search of n generations in a round-robin fashion and the cycle is completed. The process is repeated according to the LS-Interval. After the specified LS-Interval has been reached, the best individuals from all the sub-populations are concatenated into a meme which is further refined as shown in Fig. 2. The meme replaces the weakest individual in the local search population. The meme is then refined using the local search population for a given number of generations as defined by the LSI. The refined meme is then disintegrated and copied to the respective sub-populations. The refined meme replaces the weakest individuals in each of the sub-populations. Note that even if the refined meme is not improved, it replaces the weakest individuals as it may have features which will be used later in evolution. However, the best memes in the local search population are always retained. Although crossover-based local search is used as the designated



(a) Function Evaluations for evaluating the LS-Interval



(b) Success Rate for evaluating the LS-Interval

Fig. 3. The evaluation of the local search interval using the 4-bit-parity, Iris and Wine classification problems. The interval of 1 shows the highest success rate and least number of function evaluations for all three problems. (a) Function evaluations for evaluating the LS-Interval and (b) success rate for evaluating the LS-Interval.

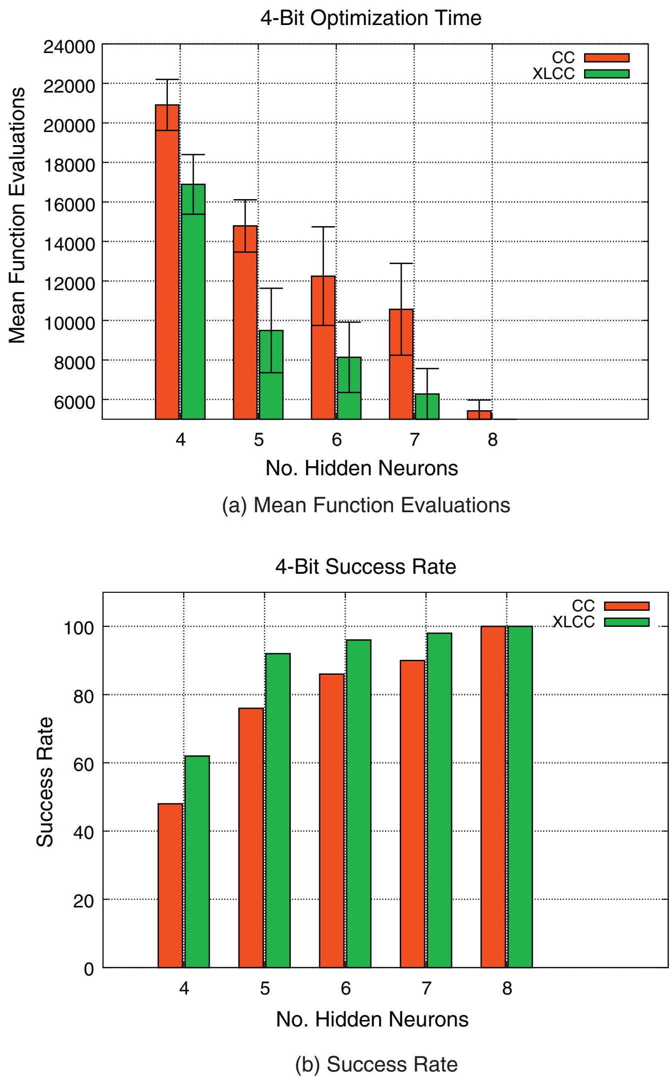


Fig. 4. The 4-bit parity problem. (a) Mean function evaluations and (b) success rate.

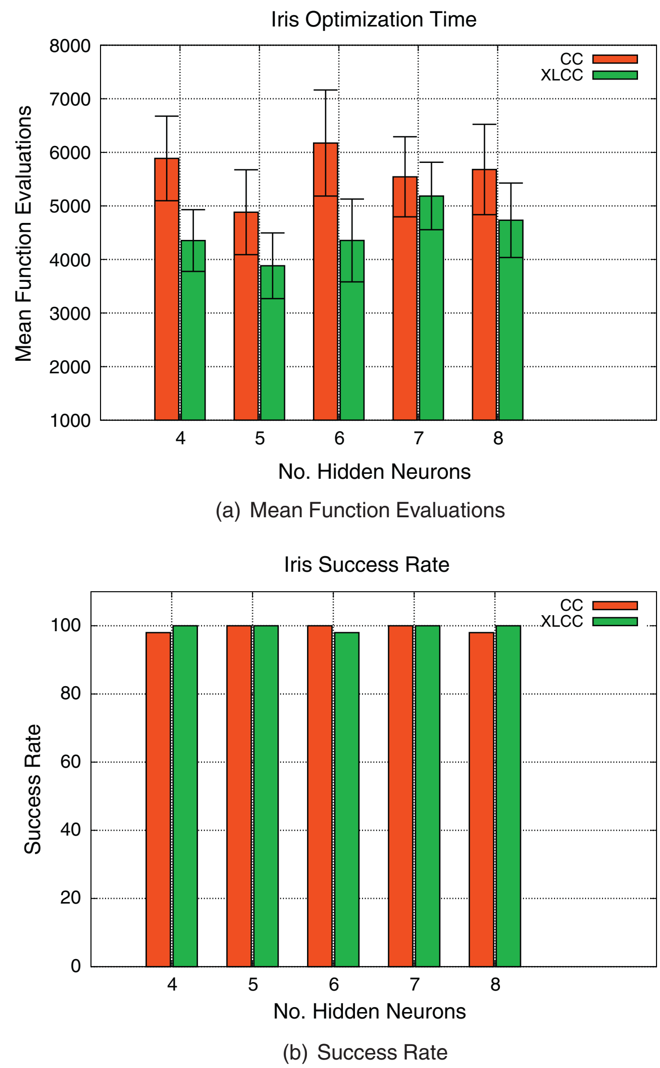


Fig. 5. The Iris classification problem. (a) Mean function evaluations and (b) success rate.

method here, the framework can employ any other local search method in principle.

The individuals in the population of the crossover-based local search are randomly initialised at the beginning of the evolutionary process. The cooperative coevolution sub-populations are initialised at the same time. The local search population is evolved according to the local search intensity, the best individual is transferred to the sub-population in cooperative coevolution. The remaining individuals in the local search population are kept and used for future evolutionary local search.

3.1. G3-PCX for crossover local search

The G3-PCX (generalised generation gap with parent centric crossover operator) algorithm is used as the EA for local search [4]. The size of the local search EA population is set to be 20% of the population size used in cooperative coevolution. We also used G3-PCX in the sub-populations of cooperative coevolution.

The details of the G3-PCX are given as follows. The generalised generation gap differs from a standard genetic algorithm in terms of selection and the creation of new individuals. In G3-PCX, the whole population is randomly initialised and evaluated similarly to the standard genetic algorithm. The difference lies in the optimisation phase where a small sub-population is chosen. At each generation, n best fit and m random individuals are chosen from the main population to make up a sub-population. The sub-population is evaluated at each generation and the evaluated individuals are added to the main population. In this way, over time, the individuals of the main populations are evaluated.

The best individual in the population is retained at each generation. The parent-centric crossover operator is used in creating an offspring based on orthogonal distance between the parents [4]. The parents are made of *female* and *male* components. The offspring is created in the neighbourhood of the female parent. The male parent defines the range of the neighbourhood. The neighbourhood is the distance of the search space from the female parent which is used to create the offspring. The genes of the offspring extract values from intervals associated in the neighbourhood of the female and the male using a probability distribution. The range of this probability distribution depends on the distances among the genes of the male and the female parent. The parent-centric crossover operator assigns more probability to create the offspring near the female than anywhere else in the search space.

3.2. Performance evaluation

The neural network optimisation time in terms of number of function evaluations and the success rate are considered to be

Table 1
Dataset information and neural network configuration.

Problem	Input	Output	Min. train. (%)	Max. time	No. of samples
4-Bit	4	1	–	30,000	16
Wine	13	3	95	15,000	178
Iris	4	3	95	15,000	150
Zoo	16	7	95	50,000	102
Ionosphere	34	1	98	15,000	351
Lenses	4	3	98	15,000	24
Heart	13	1	88	50,000	303
Cancer	9	1	95	15,000	699

the main performance measures for the method presented in this study. The success rate determines how well the particular algorithm can guarantee a solution within a specified time. A run is considered successful if a desired solution is found before the maximum time is reached in n number of experimental runs with different initial positions in the search space. The success rate reflects on robustness. The desired solution for neural network training is specified by a predefined minimum network error or minimum classification performance on the training data. The proposed algorithm is tested on different numbers of hidden neurons to explore its scalability and robustness.

The optimisation time is measured by the average number of function evaluations in n experiments. In cases where the algorithm has not converged, the maximum number of function evaluations is included in the calculation of optimisation time.

4. Simulation and analysis

This section presents an experimental study on the proposed memetic cooperative co-evolutionary framework. The G3-PCX algorithm [4] is employed in the sub-populations of cooperative coevolution and for crossover-based local search. The crossover-based local search has a population of 20 individuals. The cooperative co-evolutionary method has 100 individuals in all sub-populations. The G3-PCX algorithm uses a mating pool size of 2

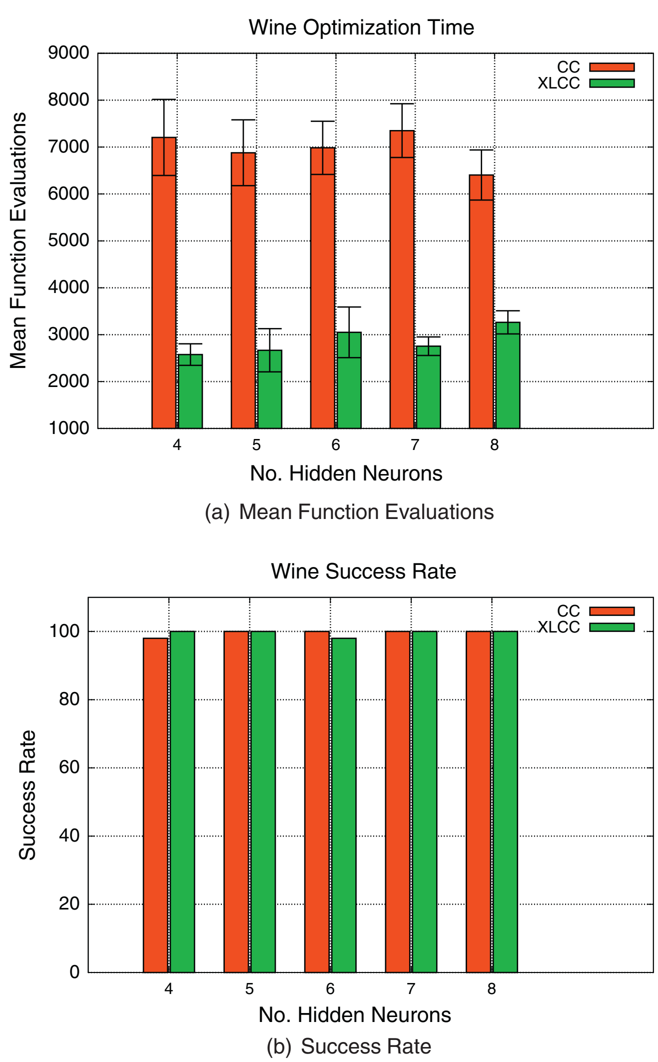


Fig. 6. The Wine classification problem. (a) Mean function evaluations and (b) success rate.

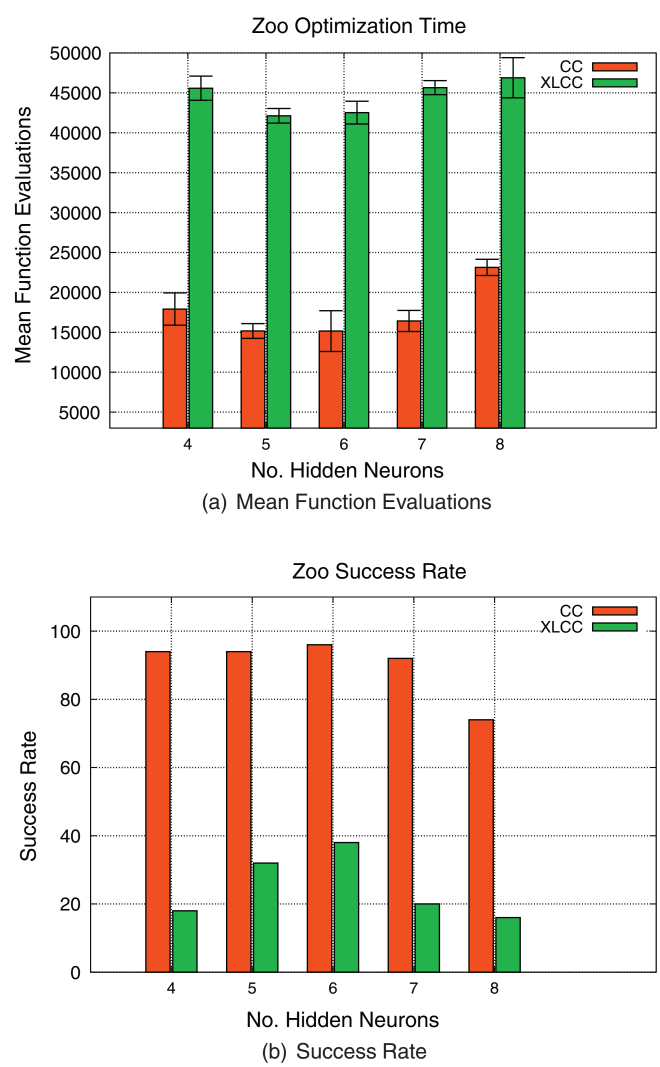


Fig. 7. The Zoo classification problem. (a) Mean function evaluations and (b) success rate.

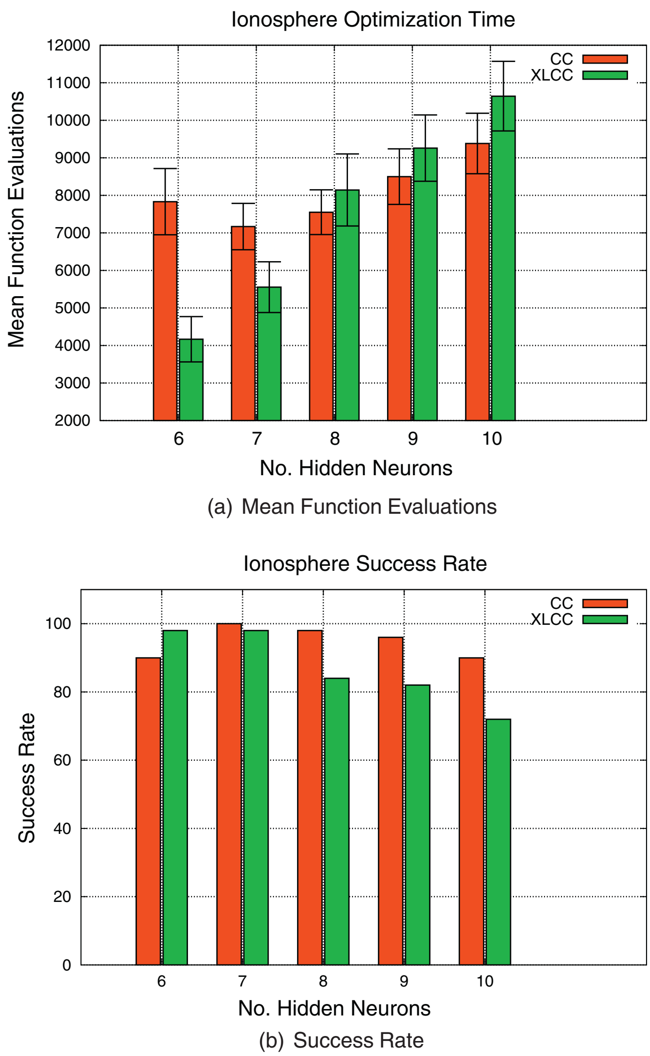


Fig. 8. The Ionosphere classification problem. (a) Mean function evaluations and (b) success rate.

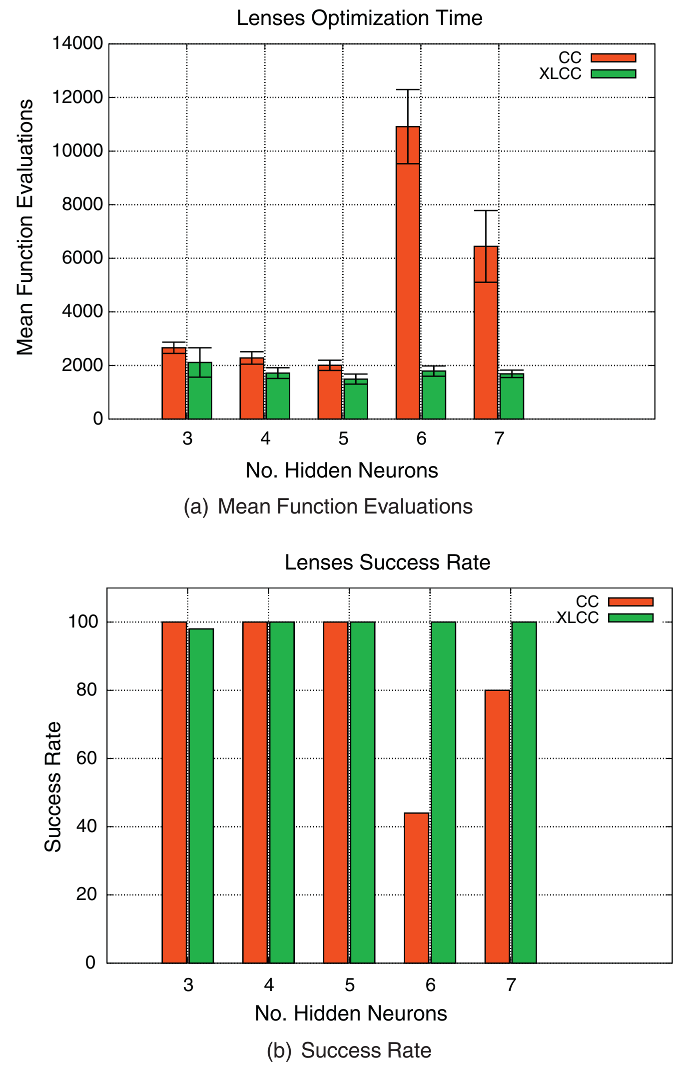


Fig. 9. The Lenses classification problem. (a) Mean function evaluations and (b) success rate.

offspring and a family size of 2 parents for all cases. This set-up has been taken from [4] and also used in previous work [2]. The sub-populations are seeded in the range of [−5, 5] in all the experiments. The cooperative coevolution framework employs the NSP encoding scheme shown in Fig. 1.

4.1. Classification problems and configuration

The *n*-bit parity problem has been used to evaluate neural network training algorithms [11]. In this work, the 4-bit-parity problem is used where an even parity is determined by the even number of 1's in the input. The *Wine*, *Iris*, *Wisconsin Breast Cancer*, *Zoo*, *Cleveland Heart Disease*, *Ionosphere*, and *Lenses* classification problems are also used to evaluate the performance of the proposed method [1].

The Wisconsin Breast Cancer dataset contains 16 missing values and is class unbalanced (65.5 % Benign and 34.5 % Malignant). The Lenses and Zoo dataset are also class unbalanced.

Table 1 gives further details of the problems with the maximum training time (max. time), minimum training (min. train.) required for termination, number of samples, number of input and output neurons used in the network. In the 4-bit-parity problem, the network is trained until the mean-squared-error goes below 1E−3. 70% of the data is used for training while the remaining 30% is used for testing in all the other problems. 50 independent runs with different initial positions in the search spaces are done for each case in the experimental set-up. The different types of problems tested here determine the number of dimensions in the optimisation problem according to the neural network topology. This enables us to measure the performance of the proposed method on problems of different levels of difficulty and scalability.

4.2. Relationship between interval and local search intensity

It is important to establish the relationship between the interval and intensity of local search in order to take full advantage for the memetic cooperative coevolution framework. The local search interval determines when to apply local refinement, i.e., after how many consecutive cycles of undergoing cooperative coevolution. We used a fixed LSI of 10 generations to evaluate the LS-Interval. We used the 4-bit parity, Iris and Wine classification problems. We employed 5 neurons in the hidden layer for these problems and used information for termination from Table 1. The results are shown in Fig. 3. A total of 50 experimental runs have been done. The best results are when the least number of function evaluations and the highest success rate is achieved. Fig. 3 shows that the local search interval of 1 for the three problems gives the best performances in terms of the least function evaluations in (a) with a high

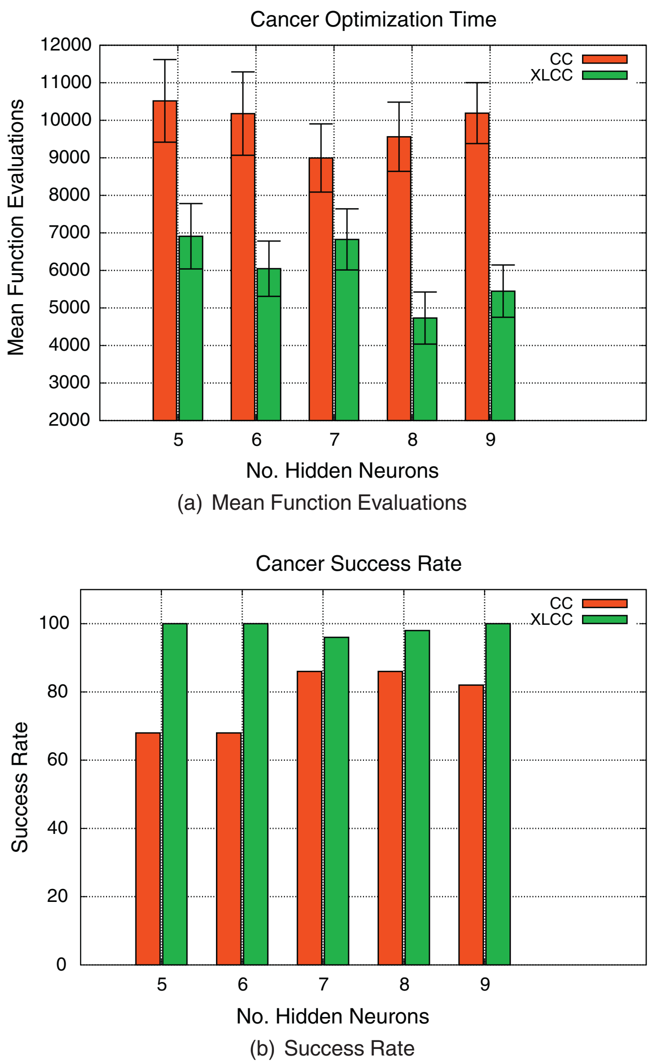


Fig. 10. The Breast Cancer classification problem. (a) Mean function evaluations and (b) success rate.

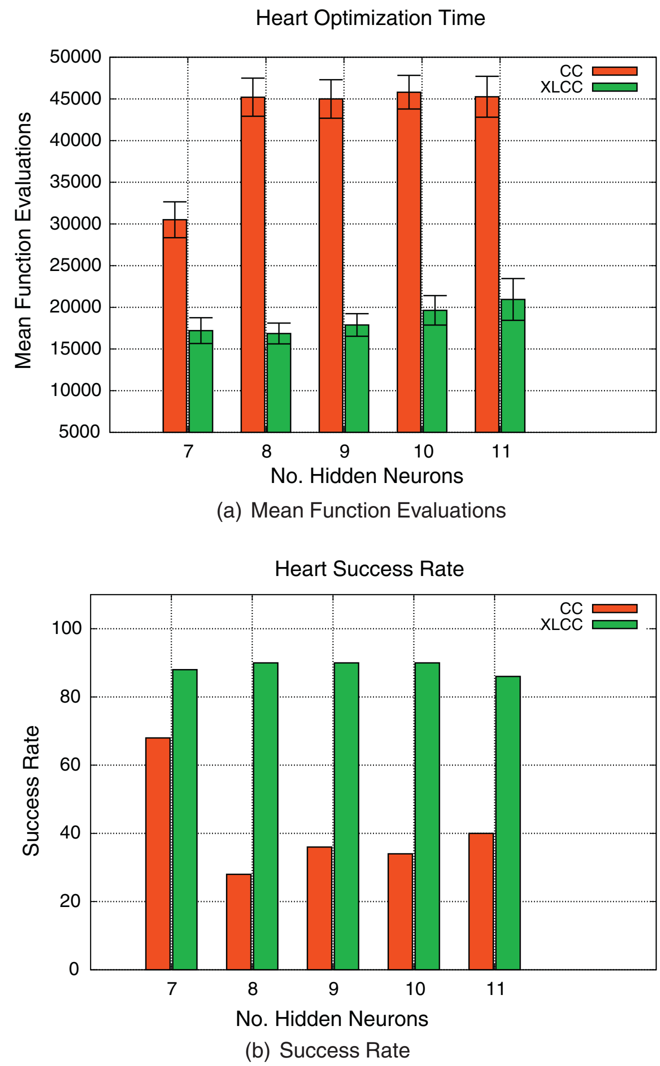


Fig. 11. The Heart Disease classification problem. (a) Mean function evaluations and (b) success rate.

success rate in (b). When the LS-Interval is increased, more diversity is introduced as the evolutionary process features evolution from the sub-populations of cooperative coevolution. It has been observed that the LS-Interval of 1 gives the best performance. One reason for this is that the local search population provides a means for co-adaptation of the sub-populations in cooperative coevolution. With the local search population, the search is carried out on individuals that represent the whole solution rather than the sub-solutions in the sub-populations. Another reason for this is the intensification process provided by the local search population.

4.3. Adaptation in local search intensity

In the previous subsection, it has been established that the local search interval of 1 gives the best performance for XLCC. It is important to use the right local search intensity. In the evolutionary process, global search is required in the initial stages and local search in later stages. Hence the local search intensity should increase during the later stages. In consideration, we propose an adaptive method for determining the local search intensity as shown in Eq. 1

$$\mathrm{LSI} = 1 + \left(\frac{t}{m} \times k\right) \quad (1)$$

where t is the number of function evaluations, m is the maximum number of function evaluations and k is a constant which specifies the maximum intensity of local search to be employed in the final stages. This heuristic ensures that the intensity of local search increases with the number of function evaluations. We used $k = 30$ for all the problems.

The heuristic proposed in Eq. 1 is used in Algorithm 1 with a local search interval of 1 and employed for training feedforward networks on different pattern classification problems whose details are given in Table 1. The results are given in Figs. 4–11 where a comparison of the memetic cooperative coevolution framework (XLCC) with the conventional cooperative coevolution (CC) framework is given. Note that the NSP encoding scheme with G3-PCX is used in both methods.

Table 2 gives the details of the generalisation performance of the two algorithms on the test data. The results shown here indicate that there is not a major difference in the generalisation performance.

The results in Figs. 4–11 show that XLCC has achieved improved performance given different numbers of hidden neurons for Iris, Wine, Heart, Cancer, Lenses, and 4-bit problems. In these problems, for all cases (hidden neurons), a higher success rate and a lower optimisation time are achieved by XLCC in comparison to CC which reflects of scalability and robustness.

Table 2
Generalisation performance on the test data.

Hidden	CC	XLCC	Hidden	CC	XLCC
Iris			**Wine**		
4	$93.67_{0.84}$	$94.90_{0.77}$	4	$92.96_{1.06}$	$93.35_{1.10}$
5	$94.30_{0.78}$	$94.80_{0.70}$	5	$94.20_{0.87}$	$93.85_{0.89}$
6	$95.75_{0.61}$	$95.56_{0.60}$	6	$94.35_{0.93}$	$93.32_{0.98}$
7	$95.85_{0.56}$	$95.75_{0.58}$	7	$93.55_{0.92}$	$94.20_{0.96}$
8	$95.51_{0.63}$	$95.75_{0.64}$	8	$92.95_{1.07}$	$93.85_{0.94}$
Lenses			**Ionosphere**		
3	$65.71_{2.24}$	$65.89_{2.11}$	6	$95.17_{0.68}$	$94.67_{0.76}$
4	$63.71_{1.97}$	$66.29_{2.21}$	7	$95.29_{0.62}$	$94.47_{0.68}$
5	$66.00_{2.36}$	$63.71_{2.27}$	8	$95.12_{0.52}$	$94.09_{1.05}$
6	$67.53_{3.21}$	$65.14_{2.40}$	9	$95.15_{0.52}$	$94.85_{0.70}$
7	$65.36_{2.40}$	$64.57_{1.98}$	10	$94.72_{0.60}$	$95.02_{0.74}$
Heart			**Cancer**		
7	$79.71_{0.96}$	$79.17_{0.69}$	5	$96.30_{0.52}$	$96.83_{0.36}$
8	$81.35_{1.15}$	$81.48_{0.48}$	6	$96.22_{0.46}$	$96.32_{0.41}$
9	$81.05_{0.96}$	$81.22_{0.67}$	7	$95.97_{0.50}$	$96.10_{0.36}$
10	$81.11_{1.01}$	$81.51_{0.46}$	8	$95.84_{0.35}$	$96.05_{0.44}$
11	$79.72_{0.92}$	$81.19_{0.44}$	9	$95.63_{0.41}$	$96.07_{0.41}$
Zoo					
4	$73.23_{1.38}$	$74.91_{2.59}$			
5	$73.92_{1.34}$	$73.79_{1.84}$			
6	$72.51_{1.17}$	$73.00_{2.16}$			
7	$73.07_{1.20}$	$71.29_{2.08}$			
8	$73.75_{1.39}$	$72.58_{2.73}$			

The results in Fig. 7 for the Zoo classification problem shows that local search (XLCC) deteriorates the performance of cooperative coevolution in all the cases. It seems that the nature of the problem requires more diversity. Good results have been obtained using cooperative coevolution alone. Co-adaptation through the local search population and intensification using crossover based local search have not been beneficial here. Intensification without proper balance of diversification through cooperative coevolution can point the search towards regions that contain local optimums. In some problems, local search is not beneficial. A similar behaviour is also seen in the Ionosphere classification problem in Fig. 8 where the performance of XLCC deteriorates for 8–10 hidden neurons. This indicates that the nature of the problem changes when more neurons are present in the hidden layer and local search is not applicable.

4.4. Discussion

The results in general show that the local search interval of 1 gives the best performance which implies that the local search has to be applied most frequently. The memetic framework has to take maximum advantage of local refinement after every cycle in cooperative coevolution in order to balance the global search feature from cooperative coevolution with local search.

The results also show that the local search is an important parameter of XLCC and its intensity is dependent on the nature of the problem. The heuristic which adapts the LSI has shown to be effective for most of the problems. The Zoo classification problem does not need local search. Local search in this problem keeps the search in local regions and adds to the computational cost.

In general, comparison of the memetic cooperative framework with standalone cooperative coevolution shows improved performance in most of the given problems in terms of optimisation time, scalability and robustness. The generalisation performance shows that there has not been a major difference in performance which implies that the memetic cooperative coevolution framework can achieve similar solution quality with better robustness and scalability. The proposed framework has the feature of efficiently utilising local search without adding to the overall computational cost in terms of function evaluations. This indicates that it is beneficial to employ local search in cooperative coevolution for training feedforward neural networks.

The local search population has also provided the means for applying co-adaptation between the sub-populations of cooperative coevolution. Co-adaptation is necessary in cooperative coevolution, especially in the case where the problem has difficulty in decomposition. It is difficult to decompose neural networks into subcomponents as the interaction between the synapses depends on the network architecture and the nature of the problem, i.e., training data. The success of the XLCC framework indicates that there is a possibility that the local search population provides a mechanism of co-adaptation. This population can also be composed of selected individuals to be exchanged with different sub-populations due to the crossover operation in the local search population.

5. Conclusions and future work

The main problem in this study was to efficiently utilise local search in the respective sub-populations of cooperative coevolution. This problem has been efficiently solved through a new framework which decomposes the locally refined solution and incorporates it into the sub-populations. The memetic framework progresses with a global search with cooperative coevolution and adapts itself by incorporating refined solutions through crossover-based local search. The memetic framework also provides the feature of co-adaptation among the sub-populations of cooperative coevolution using the local search population.

The proposed memetic cooperative coevolution framework has performed better for most of the given problems when compared to the performance of cooperative coevolution. This opens the road for further research in using other local refinement methods. In the case of training neural networks, back-propagation can also be utilised. It can replace the crossover-based local search or be used as an additional tool for local refinement. This will enable the memetic cooperative coevolution framework to incorporate gradient information from back-propagation into the evolutionary search process.

Future work can focus on the use of the memetic cooperative coevolution framework in training other neural network architectures. It can also be used for multi-objective and global optimisation. The use of other local search methods can be further investigated.

References

[1] A. Asuncion, D. Newman, UCI Machine Learning Repository, 2007, URL http://archive.ics.uci.edu/ml/datasets.html.

[2] R. Chandra, M. Frean, M. Zhang, An encoding scheme for cooperative coevolutionary feedforward neural networks, in: J. Li (Ed.), AI 2010. Advances in Artificial Intelligence. Vol. 6464 of Lecture Notes in Computer Science, Springer, Berlin/Heidelberg, 2010, pp. 253–262.

[3] R. Chandra, M. Frean, M. Zhang, C.W. Omlin, Encoding subcomponents in cooperative co-evolutionary recurrent neural networks, Neurocomputing 74 (2011) 3223–3234.

[4] K. Deb, A. Anand, D. Joshi, A computationally efficient evolutionary algorithm for real-parameter optimization, Evolutionary Computation 10 (4) (2002) 371–395.

[5] N. Garcia-Pedrajas, C. Hervas-Martinez, J. Munoz-Perez, Multi-objective cooperative coevolution of artificial neural networks (multi-objective cooperative networks), Neural Networks 15 (10) (2002) 1259–1278.

[6] N. Garcia-Pedrajas, C. Hervas-Martinez, J. Munoz-Perez, COVNET: a cooperative coevolutionary model for evolving artificial neural networks, IEEE Transactions on Neural Networks 14 (3) (2003) 575–596.

[7] F. Gomez, J. Schmidhuber, R. Miikkulainen, Accelerated neural evolution through cooperatively coevolved synapses, Journal of Machine Learning Research 9 (2008) 937–965.

[8] F.J. Gomez, Robust non-linear control through neuroevolution, Technical Report AI-TR-03-303, Ph.D. Thesis, Department of Computer Science, The University of Texas at Austin, 2003.

[9] N. Hansen, A. Ostermeier, Completely derandomized self-adaptation in evolution strategies, Evolutionary Computation 9 (2) (2001) 159–195.

[10] F. Herrera, M. Lozano, J. Verdegay, Tackling real-coded genetic algorithms: operators and tools for behavioural analysis, Artificial Intelligence Review 12 (1998) 265–319, http://dx.doi.org/10.1023/A:1006504901164.

[11] M.E. Hohil, D. Liu, S.H. Smith, Solving the *n*-bit parity problem using neural networks, Neural Networks 12 (9) (1999) 1321–1323, URL http://www.sciencedirect.com/science/article/B6T08-3XK6SSV-9/2/b500d038ee070010f0060e1033ed87fe.

[12] S.A. Kazarlis, S.E. Papadakis, I.B. Theocharis, V. Petridis, Microgenetic algorithms as generalized hill-climbing operators for ga optimization, IEEE Transactions on Evolutionary Computation 5 (3) (2001) 204–217.

[13] M. Lozano, F. Herrera, N. Krasnogor, D. Molina, Real-coded memetic algorithms with crossover hill-climbing, Evolutionary Computation 12 (3) (2004) 273–302.

[14] D. Molina, M. Lozano, C. García-Martínez, F. Herrera, Memetic algorithms for continuous optimisation based on local search chains, Evolutionary Computation 18 (1) (2010) 27–63.

[15] P. Moscato, On evolution, search, optimization, genetic algorithms and martial arts: towards memetic algorithms, Tech. rep., 1989.

[16] P. Moscato, On evolution, search, optimization, genetic algorithms and martial arts: towards memetic algorithms, Technical Report 826, Caltech Concurrent Computation Program, 1989.

[17] P. Moscato, A gentle introduction to memetic algorithms, in: Handbook of Metaheuristics, Kluwer Academic Publishers, 2003, pp. 105–144.

[18] M.A. Potter, K.A. De Jong, Cooperative coevolution: an architecture for evolving coadapted subcomponents, Evolutionary Computation 8 (1) (2000) 1–29.

[19] M.A. Potter, K.A.D. Jong, A cooperative coevolutionary approach to function optimization, in: PPSN III: Proceedings of the International Conference on Evolutionary Computation. The Third Conference on Parallel Problem Solving from Nature, Springer-Verlag, London, UK, 1994, pp. 249–257.

[20] S.-M. Soak, S.-W. Lee, N. Mahalik, B.-H. Ahn, A new memetic algorithm using particle swarm optimization and genetic algorithm, in: B. Gabrys, R. Howlett, L. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems. Vol. 4251 of Lecture Notes in Computer Science, Springer, Berlin/Heidelberg, 2006, pp. 122–129.