A Memetic Framework for 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

Abstract

Cooperative co-evolution has been a major approach in neuro-evolution. Memetic computing approaches employ local refinement to selected individuals in a population. The use of crossover-based local refinement has gained attention in memetic computing. This work proposes a cooperative co-evolutionary framework that utilises the strength of local refinement from memetic computing. It employs a crossover-based local search for refinement. The proposed framework is used for training feedforward neural networks on pattern classification problems. The results show that the proposed approach can achieve better performance than the standard cooperative coevolution framework.

Keywords: Memetic algorithm, local refinement, cooperative coevolution, and feedforward networks.

1. Introduction

Memetic algorithms (MAs) (Moscato, 1989) typically combine population-based evolutionary algorithms with local refinement which provides intensification while retaining the diversification in the search process. The meme is considered to be an individual which goes through local refinement in memetic algorithms. The search for more efficient local refinement techniques has been a major focus of study in memetic computation. There is a need to use non-gradient based approaches in local searches especially in problems where gradient information is difficult to obtain. Crossover based local search techniques have recently gained attention (Lozano et al., 2004; Molina et al., 2010). In crossover based local search, efficient crossover operators which have local search properties are used for local refinement with a population of a few individuals. They have shown promising results in comparison with other evolutionary approaches for problems with high dimensions (Molina et al., 2010).

Cooperative co-evolution (CC) divides a large problem into smaller subcomponents and solves them independently (Potter and Jong, 1994). The subcomponents are represented using subpopulations which are genetically isolated and the only cooperation takes place in fitness evaluation. Cooperative co-evolution has shown to be effective for neuro-evolution of feedforward and recurrent networks using different geneotype-phenotype mappings which determines the way individual subcomponents are built (Gomez, 2003; Gomez et al., 2008; Chandra et al., 2009).

Cooperative co-evolution has the feature of decomposing a problem using several subpopulations which provides greater diversity and increased global search features. Memetic computing provides further enhancement to evolutionary algorithms with local refinement. There has been much research in using local refinement with standard evolutionary algorithms. The success of local refinement in standard evolutionary algorithm gives the motivation of using local refinement in cooperative co-evolution.

This work proposes a Lamarckian based cooperative co-evolution framework which utilises the strength of individual learning. We choose the crossover-based local search as the method of local refinement and name the proposed method crossover-based Lamarckian cooperative co-evolution (XLCC). The proposed framework is used for training feedforward networks on classical 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 co-evolution.

The rest of the paper is organised as follows. Section II presents the background on cooperative co-evolution while Section III presents the Lamarckian cooperative coevolution framework which employs crossover-based local search. Section IV presents experimental results and section V concludes the work with a discussion on future work.

2. Background

2.1. Cooperative co-evolution

In the evolutionary process of nature, different species compete in order to survive with the given resources. The individuals of a particular group of species mate among themselves in order to produce stronger individuals. However, mating in between different species is not feasible. The cooperative coevolution framework is nature inspired where species are represented as subcomponents which are implemented as subpopulations. A major advantage of the CC framework in neuro-evolution is that it provides a mechanism for assigning genotype-phenotype mapping of subnetworks into different subpopulations.

The subpopulations in the cooperative co-evolution framework are evolved separately and the cooperation only takes place for fitness evaluation for the respective individuals in each subpopulation. The size of a subcomponent and the way a subcomponent is encoded from the problem is dependent on the problem. The way the algorithm cooperatively evaluates each subcomponent has been a major study in the CC framework. A method for estimating fitness has been proposed by Potter and Jong (Potter and De Jong, 2000). This method obtains the fitness of each individual in a subpopulation by combining it with the best individuals for the rest of the subpopulations. This method of fitness assignment has been used to train cascade networks on the two-spirals problem and has shown to learn the task with smaller networks when compared to the cascade correlation learning architecture (Potter and De Jong, 2000).

The original cooperative co-evolution framework has also been redesigned and applied in different ways. The hierarchical collaborative approach (Delgado et al., 2004) evolves the Takagi-Sugeno (Takagi and Sugeno, 1983) fuzzy model parameters on four different hierarchical levels. These levels include the level of fuzzy systems, rule-base, individual rule, and partition set. The approach was applied to function approximation and intertwined spirals problem and compared to a powerful fuzzy inference system called adaptive network-based fuzzy inference system (ANFIS) (Jang, 1993). The proposed hierarchical co-evolutionary approach showed to outperform ANFIS. Maniadakis and Trahanias proposed a hierarchical cooperative co-evolution for agent based systems (Maniadakis and Trahanias, 2006) which was used for redesigning brain inspired artificial cognitive systems. The system was embedded into a simulated robotic platform which supported environmental interaction.

Applications of cooperative coevolution include the construction of Bayesian networks for data mining (Wong et al., 2004), designing neural networks ensembles (Garcia-Pedrajas et al., 2005) and cooperative constructive method for designing neural network for pattern classification (García-Pedrajas and Ortiz-Boyer, 2007). Cooperative coevolution has shown to be very promising for large scale function optimisation problems which has been a major challenge to evolutionary computation (Yang et al., 2008).

2.2. The genotype-phenotype mapping

The genotype-phenotype mapping has been a major study in using the cooperative co-evolution framework for neuro-evolution which determines the way a subcomponent is designed. The major subcomponent design methodologies include subcomponent design on the *neuron level* and *synapse level*. The neuron level subcomponent design uses 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 (COVNET) (Garcia-Pedrajas et al., 2003) and multi-objective cooperative networks (MOBNET) (Garcia-Pedrajas et al., 2002) build subcomponents by mapping all input and output connections from the respective hidden neuron. They have been used for training feedforward network architectures. This genotype-phenotype mapping is similar to that of *enforced subpopulations* (ESP) for training recurrent neural networks (Gomez and Mikkulainen, 1997; Gomez, 2003). The neuron based subpopulation (NSP) (Chandra et al., 2009, 2010) decomposes the network to the neuron level and unlike ESP, the subcomponents do not include the outgoing weight links associated with a neuron as shown in Figure 1.

In this work, the NSP (Chandra et al., 2010) is used for training feedforward networks where each subpopulation in a layer composed of the following.

1. Hidden layer subpopulations: weight-links from each neuron in the $hidden_j$ layer connected to all $input_i$ neurons and the bias of $hidden_j$.
2. Output layer subpopulations: weight-links from each neuron in the $output_k$ layer connected to all $hidden_j$ neurons and the bias of $output_k$

Each neuron in the hidden and output layer acts as a reference point to its subpopulations.

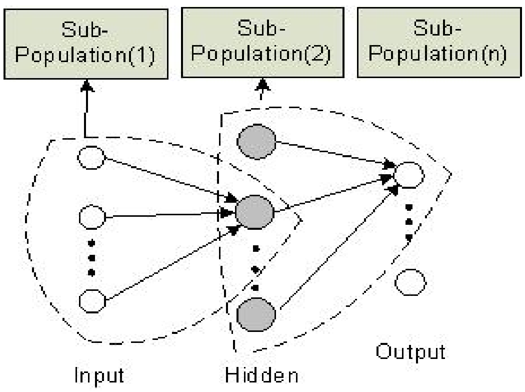


Figure 1: The NSP encoding scheme. The same encoding scheme is used in the rest of the neurons in the hidden and output layer.

3. XLCC: Crossover-based local search in Lamarckian Cooperative Co-evolution

Memetic based approaches have mainly been developed using evolutionary algorithms which have a population of individuals. In the case of building a memetic computation framework for multiple subpopulations in cooperative coevolution, we need to consider computational costs of having individual refinement for each subpopulation. The respective individuals in a subpopulation that undergo individual refinement only represents a subset of the large problem. In order to apply local refinement, the respective individual has to be concatenated with the best individuals in the rest of the subpopulations. Therefore, given n subpopulations, n local refinement would be required which would add to computational cost. Note that local refinement is applied for the *Local Serach Intensity* or duration of k iterations.

We propose a framework which efficiently takes in advantage of the local search while at the same time lowers the computational cost of having a separate local search for every subpopulation. Rather than employing a local search for each subpopulation, our proposed framework employs local search only when a *Cycle* is complete. The completion of a cycle in CC indicates that all the respective subpopulations have been evolved for a given number of generations.

Alg. 1 Framework for Lamarckian Based Cooperative Coevolution

– Encode the neural network using an appropriate genotype-phenotype mapping
– Randomly initialise all subpopulations
– Cooperatively evaluate each subpopulation

while NOT termination **do**
  for each subpopulation **do**
    i) Create new individuals using genetic operators
    ii) Place new individuals in respective subpopulation
  end for

    – Concatenate the best individuals from each subpopulation into meme M
    – Encode M into neural network
    – Local refinement on M for n iterations

  i) Decompose the refined individuals for respective subpopulation
  ii) Replace the worst individuals of the respective subpopulations with the decomposed individual
end while

The details of the proposed Lamarckian based cooperative co-evolutionary 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 subpopulation according to the respective cooperative coevolution encoding scheme (either ESP, CoSyNE or NSP (Gomez et al., 2008; Chandra et al., 2009)). The specific encoding scheme for this work is NSP which has already been discussed earlier. All the individuals of the respective subpopulation are initialised with random real values in some interval $[-a, a]$. Each individual chromosome is then concatenated with the best individuals of the rest of the subpopulations and then decoded into a neural network and evaluated as done in (Potter and De Jong, 2000). The algorithm proceeds as a standard evolutionary algorithm

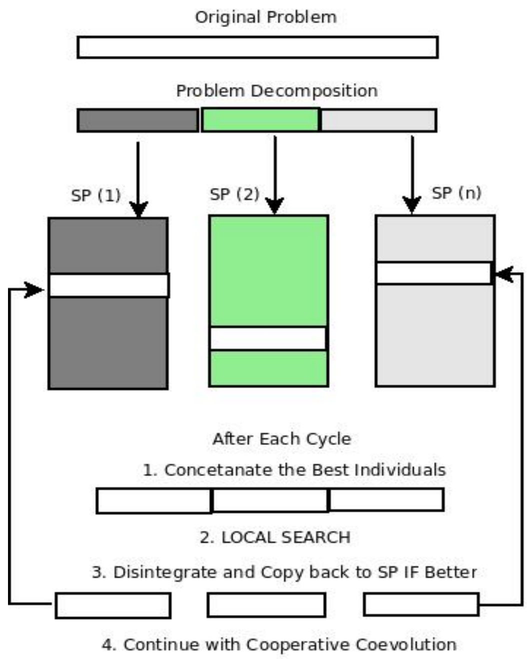


Figure 2: Proposed LCC framework to employ LOCAL SEARCH or local refinement after concatenating the best individuals from each subpopulation (SP) at the end of each cycle.

which employs genetic operators such as selection, crossover and mutation to create new offsprings for all the subpopulations. After the completion of each cycle, the best individuals from all the subpopulations are concatenated into a meme which is further refined as shown in Figure 2. The meme is then refined using crossover based local search where a population of few individuals are evolved for a given number of generations. The duration of local refinement is known as the *Local Search Intensity*. The refined meme is then disintegrated and copied to the respective subpopulations. The refined meme replaces the worst individuals in each of the subpopulations. Note that even if the refinement meme is not improved, it replaces the worst individual, as it may have features which will be used later in evolution using genetic operators. Although crossover-based local search is used as the designated algorithm for local refinement, the framework can employ any other local search method.

4. Simulation and Analysis

This section presents an experimental study on the proposed Lamarckian cooperative co-evolutionary framework.

The G3-PCX evolutionary algorithm (generalised generation gap with parent-centric crossover) (Deb et al., 2002) is employed in the respective CC frameworks and also used for crossover-based local search. The crossover-based local search has a population of 20 individuals. The cooperative co-evolutionary framework has 100 individuals in all subpopulations. These parameters were determined in trial experiments.

The G3-PCX algorithm uses a mating pool size of 2 offspring and a family size of 2 parents for all the respective CC frameworks. This set-up has been used in (Deb et al., 2002) and has shown good results for general optimisation problems. The subpopulations were initialised with random real numbers in the range of [-5, 5] in all experiments. The cooperative coevolution framework uses NSP genotype-phenotype mapping as shown in Figure 1.

4.1. Classification Problems and Neural Network Configuration

The n-bit parity are classical problems and has been used to evaluate neural network training algorithms (Hohil et al., 1999). 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* and *Iris* classification problem are also used to evaluate the performance of the proposed method (Asuncion and Newman, 2007).

For the 4-bit-parity problem, the network is training until the sum-of-squared error goes below 0.001. The network is trained until at-least 98 percent of the trained data is correctly classified in the Wine problem. The network topology configuration for each problem is also given. The 4-bit-parity problem has maximum training time of 30,000 function evaluations. The Iris and Wine classification problems have maximun training time of 10,000 function evaluations. 70 percent of the data in the wine classification problem is used for training and the remaining for testing. The neural network had 4 neurons in the hidden layer for 4-bit-parity, Iris and Wine problem.

The crossover-based local search operator is equipped with a restart scheme which keeps track on the overall error of the system. If the overall error does not decrease in five consecutive cycles, the population of crossover-based local search is re-initialised randomly. However, the strongest individual is always retained.

4.2. Local Search Intensity and Frequency

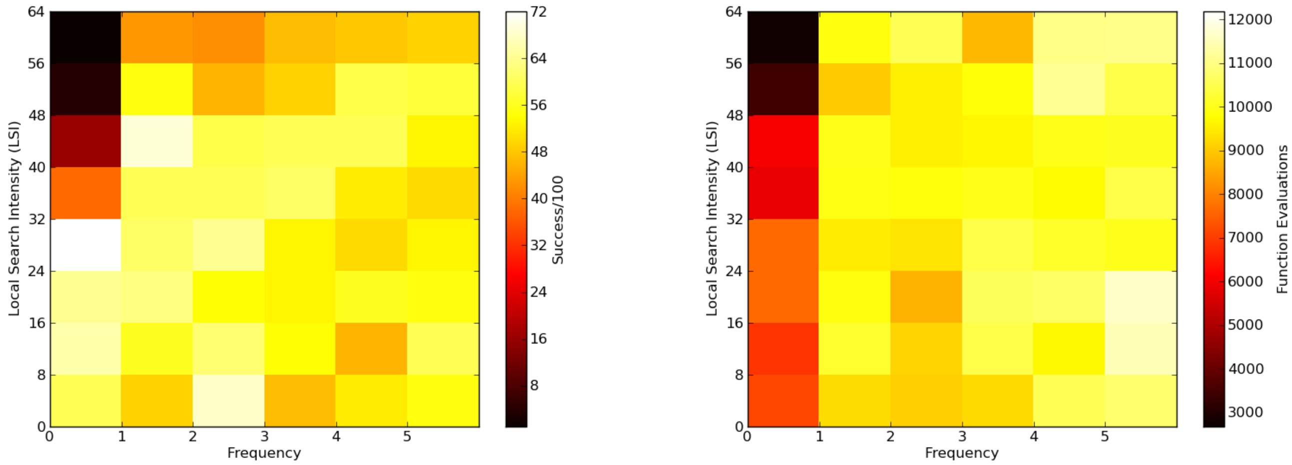
The two main parameters in the Lamarckian cooperative co-evolutionary approach are the *Local Search Intensity (LSI)* and *Frequency* of local refinement. The LSI determines how long the local refinement is done and the frequency determines when to apply local refinement, i.e, after how many consecutive Cycles of undergoing cooperative co-evolution. The best result are when the least number of function evaluations and the highest number of successful runs out of 100 experiments is achieved.

In order to evaluate the relationship between the frequency and the LSI, the results are shown as heatmaps in Figure 3(a) and 3(b) for the 4-bit-parity problem. The LSI is given by the number of generations used in the crossover-based local search method.

Figure 3(a) shows that the frequency of 1 with LSI of 32 generations show the best success rate with corresponding lower number of function evaluations as given in Figure 3(b). In some cases, the success rate is high in Figure 3(a) for frequency of 2 and 3; however, its corresponding number of function evaluations in Figure 3(b) is poor. Therefore, we choose the frequency of 1 as the optimal.

Figure 4 and 5 gives further details to the results in Figure 3 for the 4-bit-parity problem. The figures also gives the results for the Iris and Wine classification problems.

Figure 4 evaluate the behaviour of XLCC on different frequencies for the three problems. The LSI of 8 generations is used. The figure shows that the frequency of 1 gives the best results in terms of function evaluations in Figure 4(a) with higher success rates in Figure 4(b). The frequencies higher than 1 require greater number of function evaluations in Figure 4(b).



(a) Success Rate in 4-Bit-Parity Problem

(b) Function Evaluations in 4-Bit-Parity Problem

Figure 3: The heatmap shows the perfomance of XLCC on different Frequencies and Local Search Intensity (LSI) for the 4-Bit-Parity problem. The number of sucessful runs is evaluated in (a) while the number of function evaluations is evaluated in (b). The goal of XLCC is to obtain maximun success with the minimum number of function evaluations. The Frequency of 1 and LSI of 32 shows the best success rate in (a) with corresponding least number of function evaluations in (b).

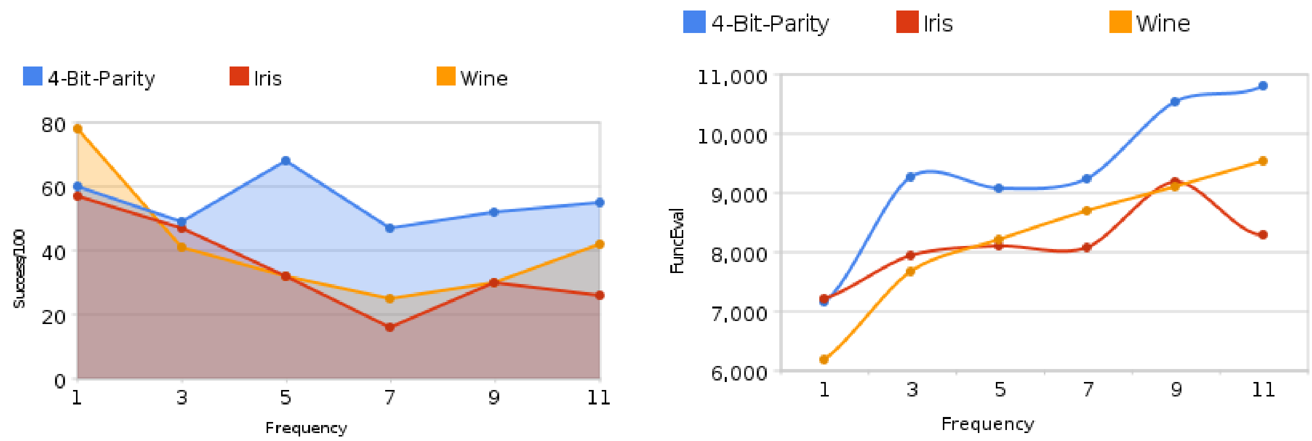
Figure 5 evaluates the behaviour of XLCC on a fixed frequency of 1 with different values for LSI. The 4-bit-parity problem shows good performance in terms of success rate in Figure 5(a) and function evaluations in Figure 5(b) for the LSI of 16-32 generations. The Wine classification problem gives good performance with LSI of 32-48 generations while the Iris classification problem has good performance with LSI of 16-32 generations.

The trend is similar in all problems where for a certain interval, good performance is achieved and later with deeper local search, poorer performance is reported. The difference in the behaviour of XLCC with different LSI values is due to the nature of the problem. In some problems, a smaller local search intensity is required, while in others, a deeper is required.

Table 1 shows a comparison of XLCC with standard cooperative co-evolution. This table consists of the best results in terms of the highest success rate from Figure 5. Note that the NSP based genotype-phenotype mapping with G3-PCX is used in both methods. The results show that performance of XLCC has improved in terms of the number of function evaluations and the success rate.

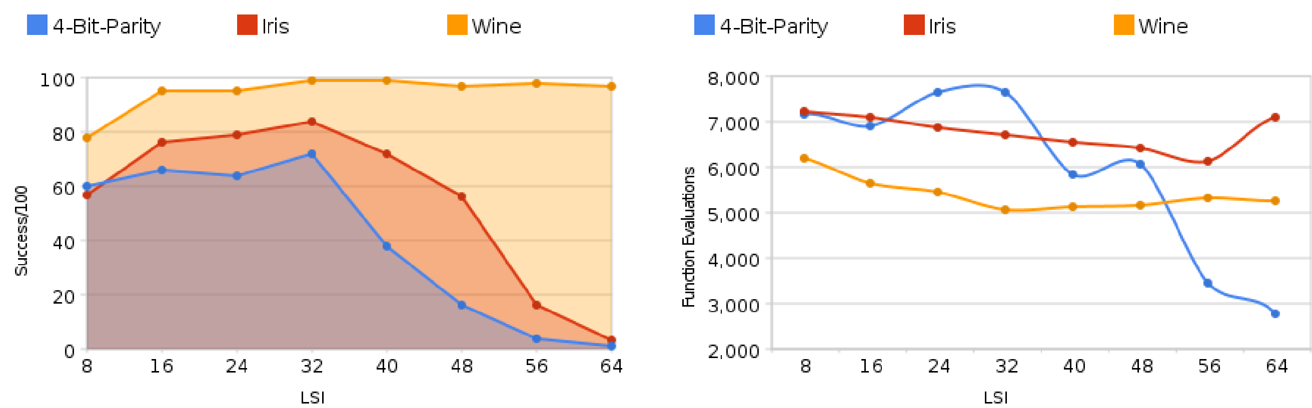
Table 1: Comparison of XLCC with CC

Problem	Method	LSI	Frequency	FuncEval	Success
4-Bit	CC	–	–	10863 $_{1093}$	55
	XLCC	32	1	7645 $_{674}$	72
Wine	CC	–	–	9408 $_{866}$	71
	XLCC	32	1	5077 $_{301}$	99
Iris	CC	–	–	8037 $_{370}$	53
	XLCC	32	1	6696 $_{282}$	84



(a) Success Rate for evaluating the Frequency (b) Function Evaluations for evaluating the Frequency

Figure 4: The evaluation of the Frequency for the 4-bit-parity, Iris and Wine classification problems. Note that the frequency of 1 shows the highest success rate and least number of function evaluations for all three problems.



(a) Success Rate for evaluating the LSI (b) Function Evaluations for evaluating the LSI

Figure 5: The evaluation of LSI for the 4-bit-parity, Iris and Wine classification problems.

4.3. Discussion

The results in general show that the frequency of 1 gives the best performance in all four problems 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 co-evolution in order to balance the global and local search.

The results also show that the local search intensity is an important parameter of XLCC and its depth on search is dependant on the nature of the problem.

In general, comparison of the XLCC framework with standard CC framework in Table 1 shows improved performance in all given problems. The success rate and the total number of function evaluations are improved. This implies that it is important to employ local search in cooperative co-evolutionary framework for training feedforward neural networks. The XLCC framework presented in this paper has the feature of efficiently utilizing local search without adding to the overall computational cost in terms of function evaluations.

5. Conclusions and Future Work

The main problem in this study was to efficiently utilise local refinement in the respective subpopulations. This problem has been efficiently solved through the proposed framework which decomposes the locally refined solution and incorporates it into the subpopulations. The memetic framework progresses with a global search through evolution of standard cooperative coevolution and as a specified time is reached (in terms of frequency of individual refinement), the algorithm adapts itself by incorporating local refinement in the subpopulations. The best refinement individuals are always added to the subpopulations even though it has not shown major improvement in solution space. This feature allows cooperative coevolution to utilise the weaker locally refined solutions in future evolution.

The proposed memetic cooperative coevolution framework has performed better for all the given problems when compared to the performance of standard 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 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.

In future work, it will be interesting to apply the proposed approach for training recurrent neural networks. Further work can be done in its application to learning long-term dependency problems. The use of other local search methods can be experimented in this framework. The same paradigm can also be extended for general global optimisation problems.

References

Asuncion, A., Newman, D., 2007. UCI machine learning repository.
URL `http://archive.ics.uci.edu/ml/datasets.html`

Chandra, R., Frean, M., Zhang, M., 2010. An encoding scheme for cooperative coevolutionary neural networks. In: 23rd Australian Joint Conference on Artificial Intelligence. Lecture Notes in Artificial Intelligence. Springer-Verlag, Adelaide, Australia, p. In Press.

Chandra, R., Frean, M., Zhang, M., Omlin, C., 2009. Building subcomponents in the cooperative coevolution framework for training recurrent neural networks. Technical Report ECSTR09-14, School of Engineering and Computer Science, Victoria University of Wellington, New Zealand.

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

Delgado, M., Von Zuben, F., Gomide, F., 2004. Coevolutionary genetic fuzzy systems: a hierarchical collaborative approach. Fuzzy Sets and Systems 14 (1), 89–106.

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

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

Garcia-Pedrajas, N., Hervas-Martinez, C., Ortiz-Boyer, D., 2005. Cooperative coevolution of artificial neural network ensembles for pattern classification. IEEE Transactions on Evolutionary Computation 9 (3), 271–302.

García-Pedrajas, N., Ortiz-Boyer, D., 2007. A cooperative constructive method for neural networks for pattern recognition. Pattern Recogn. 40 (1), 80–98.

Gomez, F., Mikkulainen, R., 1997. Incremental evolution of complex general behavior. Adapt. Behav. 5 (3-4), 317–342.

Gomez, F., Schmidhuber, J., Miikkulainen, R., 2008. Accelerated neural evolution through cooperatively coevolved synapses. J. Mach. Learn. Res. 9, 937–965.

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

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

Jang, J. S., 1993. ANFIS: Adaptive-network-based fuzzy inference systems. IEEE Trans. Systems, Man, and Cybernet. 23, 665–685.

Lozano, M., Herrera, F., Krasnogor, N., Molina, D., 2004. Real-coded memetic algorithms with crossover hill-climbing. Evol. Comput. 12 (3), 273–302.

Maniadakis, M., Trahanias, P., 2006. Hierarchical cooperative coevolution facilitates the redesign of agent-based systems. In: Proceedings of the Conference on the Simulation of Adaptive Behavior. pp. 582–593.

Molina, D., Lozano, M., Garca-Martnez, C., Herrera, F., 2010. Memetic algorithms for continuous optimisation based on local search chains. Evol. Comput. 18 (1), 27–63.

Moscato, P., 1989. On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. Tech. rep.

Potter, M. A., De Jong, K. A., 2000. Cooperative coevolution: An architecture for evolving coadapted subcomponents. Evol. Comput. 8 (1), 1–29.

Potter, M. A., Jong, K. A. D., 1994. 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, pp. 249–257.

Takagi, T., Sugeno, M., 1983. Derivation of fuzzy control rules from human operators control actions. In: Proceedings of the IFAC Symp. on Fuzzy Information, Knowledge Representation and Decision Analysis. Marseilles, France, pp. 55–60.

Wong, M. L., Lee, S. Y., Leung, K. S., 2004. Data mining of bayesian networks using cooperative coevolution. Decis. Support Syst. 38 (3), 451–472.

Yang, Z., Tang, K., Yao, X., 2008. Large scale evolutionary optimization using cooperative coevolution. Inf. Sci. 178 (15), 2985–2999.