Evolving Artificial Neural Networks for Multi-objective Tasks

Steven Künzel(✉) and Silja Meyer-Nieberg

Universität der Bundeswehr München, Werner-Heisenberg-Weg 39,
85577 Neubiberg, Germany
{steven.kuenzel,silja.meyer-nieberg}@unibw.de

Abstract. Neuroevolution represents a growing research field in Artificial and Computational Intelligence. The adjustment of the network weights and the topology is usually based on a single performance criterion. Approaches that allow to consider several – potentially conflicting – criteria are only rarely taken into account.

This paper develops a novel combination of the NeuroEvolution of Augmenting Topologies (NEAT) algorithm with modern indicator-based evolutionary multi-objective algorithms, which enables the evolution of artificial neural networks for multi-objective tasks including a large number of objectives. Several combinations of evolutionary multi-objective algorithms and NEAT are introduced and discussed. The focus lies on variants with modern indicator-based selection since these are considered as efficient methods for higher dimensional tasks. This paper presents the first combination of these algorithms and NEAT. The experimental analysis shows that the novel algorithms are very promising for multi-objective Neuroevolution.

Keywords: Neuroevolution · Evolutionary algorithms
Multi-objective · NEAT

1 Introduction

Reinforcement Learning (RL) is an important subdiscipline of Artificial Intelligence. It represents a very active field in research [1] as well as in practical applications. Artificial neural networks, or short neural networks, are often applied in the context of Reinforcement Learning. They act as controllers for the agents or robots. Here, Neuroevolution plays an important role since it allows to learn the structure and the weights of neural networks, see e.g. [2]. In many cases, however, the intended behaviour does not only depend on a single criterion, e.g. to balance two poles on a cart, but on multiple criteria, e.g. to balance two poles on a cart *and* to use as little energy as possible. Most real world problems are based on multiple (possibly) conflicting objectives, which makes it necessary to develop neural networks that consider *all* objectives, to make an algorithm applicable for practical issues. Today's quasi-standard in Neuroevolution, Stanley's NeuroEvolution of

K. Sim and P. Kaufmann (Eds.): EvoApplications 2018, LNCS 10784, pp. 671–686, 2018.
https://doi.org/10.1007/978-3-319-77538-8_45

Augmenting topologies (NEAT) is able to evolve neural networks that are adapted to only a single criterion of behaviour [2], which makes it not applicable for multi-objective tasks without further knowledge and abstraction of the problem.

Research regarding this topic is sparse: Only a few papers could be identified. Schrum and Miikkulainen [3] were the first to consider a combination of neural networks and evolutionary multi-objective algorithms: They used NEAT and NSGA-II[1] for multi-objective Neuroevolution. Their algorithm uses a not fully featured NEAT that operates with a modified version of NSGA-II [3]. In addition Schrum and Miikkulainen [5] introduced the Modular Multiobjective NEAT (MM-NEAT) which evolves modular neural networks (each module describes a behaviour) using NSGA-II and NEAT. The MM-NEAT uses all features of NEAT's framework in combination with the procedure of NSGA-II [5]. Van Willigen et al. [6] introduced the NEAT-PS, a combination of NEAT and SPEA2. The Pareto Strength approach used in NEAT-PS computes a single fitness value for all fitness functions and each individual. This makes it easy to apply in the environment of the original NEAT [6].

We were unable to find approaches that consider the more recent evolutionary multi-objective algorithms, which make use of quality indicators as the Hypervolume which allows a good approximation of the Pareto front combined with a sufficient spread of the solutions even for many objectives.

In the first part of this paper, Sect. 2, we provide the foundations of multi-objective optimization and evolutionary algorithms, additionally we give a brief overview of the NEAT algorithm. In the second part, we introduce a novel extension of NEAT, called mNEAT, which is potentially able to evolve neural networks for high-dimensional multi-objective tasks (Sect. 3). Furthermore, we present and investigate novel combinations of NEAT and the SMS-EMOA/R2-EMOA for the first time (Sect. 4). In Sect. 5 we define a multi-objective version of the Double Pole Balancing problem and provide an experimental analysis of the new algorithms. Finally, we give a summary and an outlook on future work in Sect. 6.

2 Foundations

This paper considers multi-objective optimization problems (MOPs) consisting of K objectives $f_k : \mathbb{R}^N \to \mathbb{R}$ which have to be optimized although they may be possibly conflicting [7, p. 8]. The term *optimization* may stand for the maximization or minimization of an objective function. In this paper, we focus on *minimization problems*. MOPs are hard to solve, because in many cases the K different objectives are conflicting. Here, evolutionary algorithms (EAs) are often applied. They were developed to address tasks for which, for example, no efficient algorithm is known or can be developed due to time constraints. Due to the evolutionary approach, which mimics natural evolution, EAs have a broad applicability with a relatively easy problem-specific adaptation. The field of evolutionary algorithms is also referred to as *Evolutionary Computing* [8, p. 14].

[1] An algorithm which addresses multi-objective optimization problems. See [4] for a detailed description of the NSGA-II.

2.1 Quality Indicators

To determine the quality of a solution in a MOP, multiple dimensions need to be considered. Therefore, we introduce the Pareto Dominance:

Definition 1 (Pareto Dominance). [7, p. 11] *A decision vector* $u = (u_1, \ldots, u_n)$ *dominates another decision vector* $v = (v_1, \ldots, v_n)$*, read as* $u \prec v$*, iff* $\forall k \in \{1, \ldots, K\} : f_k(u) \leq f_k(v) \wedge \exists k \in \{1, \ldots, K\} : f_k(u) < f_k(v)$.

The goal in MOPs is to identify the set of non-dominated solutions. The concept of Pareto Dominance represents a partial order. Therefore, decision vectors may exist that are not comparable. To make the solutions of a set A comparable in this case, quality indicators are necessary. They are defined as follows:

Definition 2 (Quality Indicator). [7, p. 251] *Let* A *be a vector of* H *sets* $A_1, \ldots, A_H$*. An* H*-ary quality indicator is a function* $\mathcal{I} : \Omega^H \to \mathbb{R}$*, which assigns the vector* $A = (A_1, \ldots, A_H)$ *a real value* $\mathcal{I}(A_1, \ldots, A_H)$.

We state a set A has a higher quality than another set B, if A at least weakly dominates B.[2] If a quality indicator $\mathcal{I}$ assigns a value $\mathcal{I}(A) \geq \mathcal{I}(B)$ under the condition that $A \preceq B$ (and vice versa), $\mathcal{I}$ is called *Pareto Compliant* [7, p. 253].

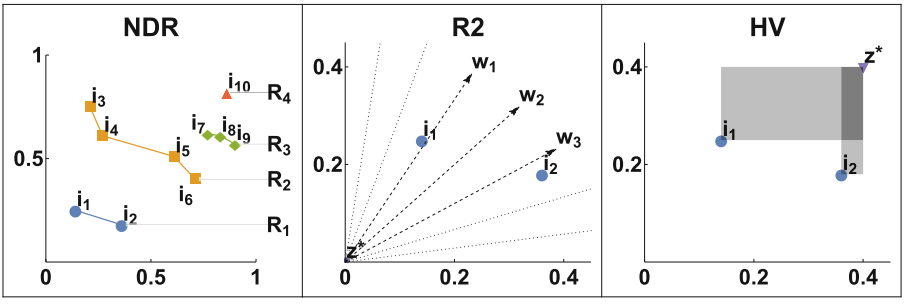

Fig. 1. Example for the quality indicators Nondominated Ranking (left), R2-Indicator (center) and Hypervolume (right). Consider a two-dimensional MOP and a set A consisting of ten individuals i_1 to i_{10}. The R2-Indicator and the Hypervolume do only focus on i_1 and i_2 in this example. The reference point is denoted by z^*.

In the following, we provide a brief overview of three important quality indicators in the context of this paper. Figure 1 gives an example for a set $A = \{i_1, \ldots, i_{10}\}$ for each of the following quality indicators:

Nondominated Ranking. The Nondominated Ranking [4] orders the solutions of a set A using the Pareto Dominance. Therefore all solutions are assigned into ranks R_1 to R_H. Let $1 \leq i < j \leq H$ then $\forall b \in R_j \, \exists a \in R_i : a \prec b$. Solutions of

[2] See [7, Table 5.2, p. 244] for an overview of the dominance relations.

equal rank are incomparable with respect to Pareto Dominance. In Fig. 1 the individuals i_1 and i_2, which dominate all remaining individuals, are assigned to R_1. The individuals i_3 to i_6, which are dominated by i_1 and i_2 each, but dominate the individuals i_7 to i_{10}, are assigned to R_2 and so on.

R2-Indicator. The R2-Indicator is based on the standard weighted Tchebycheff utility function, where z is a solution, z^* an (ideal) reference point and $\lambda = \{\lambda_1, \ldots, \lambda_K\} \in \Lambda$ is a weight vector [9]:

$$u(z) = -\max_{i \in \{1,\ldots,K\}} \{\lambda_i |z_i^* - z_i|\} \tag{1}$$

Λ is a set of (usually) uniformly distributed weight vectors over the weight space [10]. The R2-Indicator returns the averaged sum of the minimum distances for all $a \in A$ in any dimension for each weight vector $\lambda \in \Lambda$:

$$R2(A) = R2(A, \Lambda, z^*) = \frac{1}{|\Lambda|} \sum_{\lambda \in \Lambda} \min_{a \in A} \left\{ \max_{i \in \{1,\ldots,k\}} \{\lambda_i |z_i^* - a_i|\} \right\} \tag{2}$$

A lower R2 value means that the set's solutions are located closer to the reference point. The contribution of a single solution $a \in A$ to the R2 value is computed by $R2(A \setminus \{a\}) - R2(A)$ [10]. In Fig. 1 there are three weight vectors $w_1 = \left(\frac{3}{8}, \frac{5}{8}\right)$, $w_2 = \left(\frac{1}{2}, \frac{1}{2}\right)$ and $w_3 = \left(\frac{5}{8}, \frac{3}{8}\right)$. Each assigns a certain importance to each dimension. For each weight vector, the individual's contribution is increased, which is nearest to the reference point z^* with respect to the vector's weights. The following individuals are selected: i_1 for w_2 and i_2 for w_1 and w_3. In the above case, considering only i_1 and i_2 their total R2 contribution is 0.062 respectively 0.007. Thus i_1 is evaluated as better than i_2 concerning the R2-indicator.

Hypervolume. The Hypervolume (HV) defines the volume of the objective space that is covered by the solutions of a set A and a reference point z^*. The Hypervolume is defined as follows:

$$HV(A) = HV(A, z^*) = \left\{ \bigcup_{a \in A} vol(a, z^*) \right\} \tag{3}$$

where $vol(a, z^*)$ denotes the volume of the space bounded between a and z^* [7, p. 260]. The larger the value of $HV(A)$ the more space is covered by A, hence the solutions in A are closer to the ideal point (in case that z^* is the nadir point, e.g. $z^* = (1, \ldots, 1)$ for normalized minimization problems) [11]. The contribution of a single solution $a \in A$ to the HV value (the space that is *only* covered by a) is computed by $HV(A) - HV(A \setminus \{a\})$ [12]. In Fig. 1 the area, dominated exclusively by i_1 or i_2 is shaded in light gray. The dark gray area is dominated by both individuals. In this example it is obvious that the area exclusively dominated by i_1 is larger than i_2's area, therefore i_1 is to prefer over i_2 with respect to the dominated Hypervolume.

2.2 Evolutionary Multi-objective Algorithms

To address MOPs, evolutionary multi-objective algorithms (EMOAs) can be used. The class of EMOAs is also referred to as multi-objective evolutionary algorithms, short MOEAs. Well-known examples include the NSGA-II [4] and the SMS-EMOA [13]. Additionally, Trautmann et al. [14] introduced the R2-EMOA. We have selected the SMS- and the R2-EMOA for further investigations, because both have been shown to have a high performance while they do only require the user to set the population size μ. The SMS-EMOA is able to deal with an arbitrary large number of fitness functions K and uses two criteria for sorting a population's individuals: first all individuals are sorted into ranks using fast nondominated sorting of the NSGA-II [4]. If the (worst) rank R_H contains more than one individual, the Hypervolume contribution of each individual in R_H is computed to make these individuals comparable (survivor selection). Note that there exist different variants of SMS-EMOA using the Dominance count as primary selection criterion [13]. The R2-EMOA works similar to SMS-EMOA, the only difference is that the R2-EMOA relies on the R2-Indicator as the secondary criterion instead of the Hypervolume [14].

As already stated, both, the R2-EMOA and the SMS-EMOA will be considered to derive multi-objective variants of the NEAT approach. But first, the next section provides the details of the original method.

2.3 NeuroEvolution of Augmenting Topologies

The NeuroEvolution of Augmenting Topologies algorithm (NEAT) was introduced in [2]. It addresses the evolution of neural networks for performing single objective tasks. Therefore it depends on a node based encoding, where neural networks are described by a list of links, which each contains information about the connected nodes (or neurons) [2, p. 34f.]. NEAT relies on three basic principles:

Historical Markings. For the crossover of two networks, their structure has to be considered. NEAT addresses this by tracking each structural mutation (new link or neuron) with an Innovation ID. If a mutation occurs twice or more, it is always assigned the same Innovation ID. Thereby two networks can be combined without any structural analysis, the equal parts of two networks can be determined by their Innovation IDs, likewise their unequal parts [2, pp. 36–38].

Speciation. To protect innovation, NEAT sorts the networks of a population into niches of similar networks, called species. Thus networks do only have to compete against the other members of their species, and may evolve a competitive structure. Additionally it keeps the genomes as small as possible: As long as the fitness of smaller networks is comparable to the fitness of other, larger networks, the smaller networks remain in the population and thus are not unnecessarily replaced by larger networks [2, pp. 38–40].

Complexification. Every network in NEAT starts with the same minimal topology: each input neuron is directly connected to every output neuron (with random link weights), there are no hidden neurons, only an optional bias

neuron. By structural mutation the networks grow incrementally and only well performing ones will survive. Thereby NEAT needs to search a minimal number of weight dimensions and is able to find well performing networks very quickly, additionally this prevents the networks' structures from growing unnecessarily large [2, p. 40f.].

NEAT is a powerful algorithm which is able to outperform other algorithms for evolving neural networks like Cellular Encoding [15], Symbiotic Adaptive Neuroevolution [16] and Enforced Subpopulations [17] in several experiments [2, pp. 44–49]. As it has been shown, NEAT is capable to evolve neural networks for single objective tasks, but how can Neuroevolution be used to address multi-objective problems?

3 Multi-Objective NEAT: A First Approach

In the real world, there exist many tasks that are suitable to be controlled by neural networks, but depend on more than one criterion. The NEAT approach allows to evolve neural networks considering a single objective, but for problems with more than one, NEAT has to fall back to using a weighted fitness function (scalarisation). The drawback of weighted fitness functions is that the user has to determine the weights a-priori and thus needs to know the importance of each goal *before* optimization [8, p. 196]. To avoid this and other disadvantages of scalarisation, we introduce a novel multi-objective version of NEAT, called Multi-Objective NEAT (mNEAT) as the foundation of our research. The mNEAT is an indicator-based algorithm and utilizes the R2-Indicator for quality assessment. Its main procedures are described in the following.

3.1 Procedure of mNEAT

The novel approach starts with an initially random population $\mathcal{P}_t$ (for time $t = 0$) consisting of μ minimal networks. The mNEAT is based on speciation. Here, the user defines the target number of species and mNEAT automatically adjusts the speciation threshold (the maximum difference of two networks of same species) according to the networks' difference. This step is executed at the beginning of every epoch. Note that mNEAT provides a general selection and sorting procedure, that is, the outer loop of an EMOA. It assumes that the neural networks in the population ($p \in \mathcal{P}_t$) have been evaluated and assigned a fitness $f(p) = (f_1(p), \ldots, f_K(p))$ before each epoch.

The networks in $\mathcal{P}_t$ are sorted depending on their R2 contribution with respect to the population $\mathcal{P}_t$. All networks are assigned to the corresponding species s, which results in a set of species S. Already existing network-to-species mappings are removed in advance. The first network assigned to a species is chosen as its representative. After speciation, the fitness $f(s) = (f_1(s), \ldots, f_K(s))$ of each species $s \in S$ is computed (determined by the species' members' fitness): The approach uses fitness sharing to keep species as small as possible

and to maximize the diversity otherwise. Therefore, the species' fitness is computed by $\forall k \in \{1, \ldots, K\} : f_k(s) = \sum_{p \in s} f_k(p)$. When used for minimization problems, large species are assigned a comparatively large fitness with respect to smaller species. A large species has only then the chance to compete with smaller species if it contains much better networks than the smaller species. In a maximization problem, the fitness could be exponentiated with -1 for example in order to penalize larger species. The species' fitness is normalized between 0 and 1 for each fitness function, which makes the fitness functions comparable to each other. Finally the R2 contribution of each species $s \in S$ with respect to S is computed: The number of solutions, offspring(s), the species s is allowed to create, is proportional to its R2 contribution:

$$\text{offspring}(s) = \frac{R2(s, S, \Lambda, z^*)}{\sum_{t \in S} R2(t, S, \Lambda, z^*)} \left(\mu - |S|\right). \tag{4}$$

Every epoch $\mu - |S|$ offspring networks are created by crossover and mutation. Each species internally performs parent selection by using Stochastic Universal Sampling[3]. The mNEAT approach employs the variation operators that were defined in NEAT [2]. All newly created networks enter the next generation's population $\mathcal{P}_{t+1}$. Additionally, the representative (i.e., best network) of each species $s \in S$ will be part of $\mathcal{P}_{t+1}$. All other networks are discarded.

3.2 Variations of mNEAT

First experiments with mNEAT have shown that it is able to find good solutions fast but that the final Pareto front is typically only sparsely populated by good solutions because many of these are discarded during evolution. To improve the performance of mNEAT, we introduced and investigated several variations: **(1)** an *archive* of best individuals [7, p. 14], **(2)** using a *steady state* population model [8, p. 80] and **(3)** *reducing the replacement rate* in each epoch (not discarding all networks, except the species' leaders, but preserving a part of the other (mutated) networks for the next generation).

Our initial idea of the cooperation of these variations was as follows: The **archive** (Variation 1) records the progress of the algorithm and keeps a list of the best networks that have been found during the optimization. This allows the

[3] Stochastic Universal Sampling (SUS) behaves similar to the Roulette Wheel Selection (RWS), where each individual gets assigned a hole on a one-armed roulette wheel, the hole's size depends on the individual's fitness compared to all individuals' fitness. The roulette wheel is spun once and an individual is selected then. SUS uses an equally-spaced λ-armed roulette wheel to select λ different individuals at once instead of spinning the wheel λ times. The better an individual's fitness, the better it's chance to be selected as parent [8, p. 84]. Every individual (even the worst of the population) has a nonzero chance of being selected [8, p. 81f.]. The advantage of SUS over RWS is that a set of λ unique individuals can be selected at once. Using RWS for selecting $\lambda > 1$ individuals would require λ executions and a mechanism to avoid the same individual being selected twice.

mNEAT to explore arbitrary new networks without losing the progress already made. Using a **steady state** population model (Variation 2) results in always taking over the best networks of $\mathcal{P}_t$ and the offspring into $\mathcal{P}_{t+1}$. Without this variation mNEAT would only keep the species' representatives and discard any other networks. By Variation 2, the mNEAT does not discard promising networks any more, only because their fitness is not at its maximum at timestep t due to lack of evolution. Finally, the **reduction of the replacement rate** (Variation 3) allows younger networks to evolve and maximize their fitness. By doing so, less networks are newly generated but more existing networks are mutated. Thus *exploitation* plays a bigger role instead of a pronounced utilization of *exploration* [8, p. 41f.]. Preliminary experiments (Multi-objective Double Pole Balancing experiment with all eight possible combinations of these variations) have shown that the Variations 2 and 3 do only have a small influence on the algorithm's performance. Using mNEAT only with Variation 1 (archive) provides the best results with respect to the Pareto front. This behaviour is caused by the archive, which always saves the best known solutions and returns these as the finally known Pareto Front. Due to space restrictions we do not provide details concerning the experiment and its results in this paper. In future work further variations of the mNEAT will be investigated to create an even better performing algorithm.

4 NEAT as the Foundation of Evolutionary Multi-objective Algorithms

We already provided a brief description of the two efficient evolutionary multi-objective algorithms *SMS-EMOA* and *R2-EMOA* with indicator-based selection. Since these algorithms are capable of optimizing solutions in multiple dimensions simultaneously, they should be suitable for adapting the behaviour of neural networks. As both algorithms are designed as general frameworks and not specifically to work with neural networks, we introduce the following extensions, similar to Schrum and Miikkulainen [3,5] and van Willigen et al. [6]:

For giving SMS- and R2-EMOA the ability to evolve neural networks, we base them on the framework that NEAT provides: We use the same genetic encoding of neural networks, the Innovation IDs and variation (crossover and mutation) operators of NEAT and implement those for an SMS- and R2-EMOA. Following this approach, we combine the power and efficiency of SMS-EMOA and R2-EMOA for multi-objective optimization and NEAT for evolving neural networks. This results in an algorithm Multi-Objective NEAT-Indicator Based, short mNEAT-IB that is capable of evolving neural networks for multi-objective tasks quite efficiently using different components for selection. Beside the selection mechanisms used in R2-EMOA and SMS-EMOA, we apply and investigate two indicator (only) based as selection components. Table 1 gives an overview over these variations and additionally provides information about their computationally complexities.

Table 1. Variations of the mNEAT-IB using different combinations of sorting criteria. K denotes the number of objectives and μ the population size. Λ defines the set of weight vectors used for the R2-indicator. The computational complexities are based on the following values: NDR: $O\left(K\mu^2\right)$ [4], HV (contribution of individual): $O\left(\mu \log \mu + \mu^{\frac{K}{2}+1}\right)$ [11], R2 (contribution of individual): $O\left(K\mu\,|\Lambda|\right)$ [18].

	Sorting		Computational complexity	
Variation	Primary	Secondary	Minimum	Maximum
1	NDR	HV	$O\left(K\mu^2\right)$	$O\left(K\mu^2 + \mu \log \mu + \mu^{\frac{K}{2}+1}\right)$
2	NDR	R2	$O\left(K\mu^2\right)$	$O\left(K\mu^2 + K\mu\,|\Lambda|\right)$
3	HV			$O\left(\mu \log \mu + \mu^{\frac{K}{2}+1}\right)$
4	R2			$O\left(K\mu\,|\Lambda|\right)$

In the following, we provide the details of the novel neuroevolutionary approach: The mNEAT-IB creates an initial population of μ minimal networks (with randomized weights). The population is sorted by the first and, optionally second sorting criterion. Then λ (instead of strictly one like in SMS-EMOA [13]) new networks are created by using NEAT's variation operators which then are added to $\mathcal{P}_t$. The parameter λ needs to be defined by the user. A large number of offspring λ may lead to a faster convergence of the population towards the Pareto front. However, this may cause the algorithm to miss potentially good networks by replacing them too quickly. For giving younger networks a chance to evolve, mNEAT-IB provides a small fitness-boost to younger networks, while it slightly penalizes the fitness of elder ones. The 2λ parents for crossover are selected using Stochastic Universal Sampling from the sorted population $\mathcal{P}_t$. The resulting intermediate population has a size of $\mu + \lambda$ and has to be reduced by λ networks. Because the λ new networks require fitness values before the reduction, the epoch ends here and the new networks are evaluated. At the beginning of the next epoch the population $\mathcal{P}_t$ is sorted as described above. The worst λ networks are removed from $\mathcal{P}_t$ which decreases its size to μ. The remainder are already sorted and the mNEAT-IB continues with the selection of 2λ parents for the next generation. The algorithm terminates if the stopping criterion is met.

Due to the exponential growth of the Hypervolume-computation's runtime, the R2-indicator is to be preferred for a larger number of objectives ($K \geq 4$). Its runtime heavily depends on the number of weight vectors applied. Here, Brockhoff et al. [10] discuss the optimal number of weight vectors for $K = 2$. Further investigations are necessary to give a suggestion for the case $K > 2$. We use a default of 100 weight vectors for our experiments described in Sect. 5.

5 Experimental Analysis

To compare the performance of our novel algorithms and variations, we conducted a series of experiments using a variant of the Double Pole Balancing

problem. This section provides the experimental set-up and discusses the experimental findings. First we introduce a multi-objective version of the Double Pole Balancing problem before we describe the design of the experiments. The last part of the section summarizes and discusses the results.

5.1 A Multi-objective Double Pole Balancing Problem

The Double Pole Balancing problem describes the following task: Given a **cart** c with mass c_m (= 1 kg) on which two **poles** p_1 and p_2 are mounted using a hinge. Each pole p has a length p_l (= 1 m/0.1 m) and a mass p_m (= 0.1 kg / 0.01 kg). At time $t = 0$ both poles are poised, inclined in angles p_{1_α} (= 0°) and p_{2_α} (= 1°). Left by themselves, the two poles would fall to the left or right side over the time t. If one or both poles' inclination exceeds a given maximum angle γ (= 36°), the experiment has failed and the time t is stopped. The cart is positioned at the centre of a track with length L (= 4.8 m). The experiment's target is to keep the two poles balanced by moving the cart to left or right without leaving the track for a given amount of time T (= 10.000 units). The cart is controlled by a neural network which has to be evolved using our proposed algorithms. The networks consider the following input parameters: The position c_{pos} and velocity c_v of the cart and the angles and rates of fall of the poles p_1 and p_2. The networks' output is a value which determines the force that affects the cart (direction and strength) in the next timestep. The Double Pole Balancing problem was used by Stanley [2] for comparing the performance of the single-criterion NEAT to other neuroevolutionary algorithms and ablations of the original NEAT. It is an enhanced version of the Pole Balancing problem [16] which became too simple to solve for modern algorithms to be a measure for comparison [2]. The Double Pole Balancing problem is typically used to evaluate the performance of algorithms for Reinforcement Learning.

We transform this problem into a multi-objective optimization problem using the following fitness functions: **(1)** The first fitness function f_1 describes how long the controller x is capable of keeping the poles p_1 and p_2 balanced. **(2)** Additionally the number of directional changes (per time unit) r of the cart c is counted. The reason is that for each change of direction the cart has to be slowed down to $c_v = 0$ and then be accelerated in the other direction – this consumes much more energy than the movement with constant speed [19]. This results in the number of directional changes being proportional to the energy consumption of the cart. **(3)** Finally the cart's position c_{pos} relative to the centre of the track $L/2$ is considered. Therefore the average distance per timestep f_3 is computed – a larger value of f_3 means a greater distance that is covered by the cart and that results in a higher energy consumption. Additionally this leads to a lower distance to one end of the track, which increases the probability of the cart to leave the track. We introduced f_3 due to the fact that a controller can achieve a good fitness value for f_2 with the following behaviour: If the controller moves the cart from the very left to the very right end of the track continuously, the cart executes very few directional changes but always travels a distance of

nearly L between each. This would increase the energy consumption and makes f_3 necessary.

The three fitness functions read as follows, where the parameter c_{pos_t} describes the position of the cart on the track at time t:

$$\begin{array}{ll} f_1(x) = T - t & \forall x \in \Omega : 0 \leq f_1(x) \leq T \\ f_2(x) = \frac{r}{t} & \forall x \in \Omega : 0 \leq f_2(x) \leq 1 \\ f_3(x) = \frac{1}{t}\sum_1^t \left| \frac{L}{2} - c_{pos_t} \right| & \forall x \in \Omega : 0 \leq f_3(x) \leq \frac{L}{2} \end{array} \tag{5}$$

5.2 Experiments and Statistical Analysis

All algorithms under consideration have control parameters that strongly influence the performance. Therefore, we first conducted preliminary experiments with a meta-EA in order to identify suitable parameter settings. The best were investigated more closely in the context of the analysis described in this paper. Due to space restrictions we do not show the configurations in this paper, but will offer these in a technical report. We conduct two different types of experiments which are repeated 120 times each:

Average Number of Evaluations. In this experiment we test an algorithm's ability to find a solution of predefined minimum fitness within a given amount of maximum evaluated networks. For each repetition, we record the number of evaluations needed to find an individual of desired fitness. At the end of the experiment, we compare the average number of evaluations and the success rate of each algorithm. Finally, we investigate the algorithms' results for statistically significant differences.

Mean Fitness. The second experiment tests an algorithm's ability to find a "good" Pareto front. Therefore each algorithm will be executed for a predefined number of evaluated networks with the final Pareto front (last generation or archive) being stored. All solutions of all Fronts are combined (one Pareto front per repetition) and used to compute the following quality indicators: ϵ, *Spacing*, *generational Distance*, *Inverted generational Distance*, *Inverted generational Distance Plus* and *Hypervolume*. This gives us the ability to evaluate the quality of the Pareto fronts that have been found by the algorithms. We show the averaged results of each algorithm for each performance measure and finally analyse the results for statistically significant differences.

For the statistical comparison of the algorithms (with the predefined configuration) we use the procedure described by Calvo and Santafé [20] using the R-library *scmamp*[4]. First we determine if there is a statistically significant difference between any two algorithms (Friedman-Test [21]). If a difference has been ascertained we determine the pairwise difference between the algorithms (Friedman Aligned Ranks test [22]). The determined p-values are adjusted (Shaffer's algorithm [23]) and then evaluated.

[4] R-library for comparing algorithms: https://cran.r-project.org/web/packages/scmamp/index.html.

Average Number of Evaluations. Table 2 shows the results of the Double Pole Balancing experiment with velocities. We investigate the average number of evaluated networks until an algorithm found a network x^* for which $f_1(x^*) = 0$, $f_2(x^*) \leq \frac{1}{20}$ and $f_3(x^*) \leq \frac{1}{5}$. This calls for a neural network that balances the poles of the cart for at least T timesteps and performs a change of direction at most every 20^{th} timestep. Additionally it is not allowed to stay away from the centre of the track more than 20 cm on average. We were looking for a safe and energy-efficient controller for the cart.

Table 2. Number of evaluated networks until the algorithm found a network of the desired fitness in the Double Pole Balancing experiment with velocities (Average Number of Evaluations) averaged over 120 repetitions. A repetition was not successful, if no network x^* has been found within 25,000 evaluated networks.

Algorithm	Variation	Mean	+−	Success rate
mNEAT	Original	10,789	7,497	0.86
	Archive	10,733	7,870	0.85
mNEAT-IB	NDR + HV	14,150	8,295	0.7
	NDR + R2	9,946	7,836	0.85
	HV	11,180	9,537	0.72
	R2	8,272	6,050	0.92

The results in Table 2 show that the mNEAT was able to find a suitable network x^* in 85% of all repetitions. Note that the mNEAT (without variation) and the mNEAT with archive behave identical in this type of experiment[5] – this can be observed in both variants' results, which are nearly equal in this experiment. The mNEAT-IB (R2) shows the best results with respect to number of evaluations (8,272) and success rate (92%). On the other hand the mNEAT-IB (NDR + HV) evaluated 14,150 networks until it found a network x^* and thus performed worst (even statistically significant for $\alpha = 0.05$). It has to be investigated whether the quality indicators in mNEAT-IB (NDR + HV) are not suitable for evolving neural networks or this variation has not been configured properly by the meta-EA. To summarize, all algorithms were capable of finding an energy efficient controller for the cart in at least 70% of all repetitions with a maximal number of 25,000 evaluations. Most of the algorithms even exceeded a success rate of 85%.

[5] The only difference between the original mNEAT and the mNEAT with archive is that the latter saves the best solutions ever found in an archive. Because the archive is only kept as a "second population" and finally returned as Pareto front, there is no difference in both variants' behaviour in the Average Number of Evaluations experiment.

Table 3. Quality indicator values of the algorithms and variations using the previously determined parameter settings for the Double Pole Balancing experiment with velocities (Mean Fitness). (first row = mean value, second row = standard deviation)

Algorithm	Variation	ϵ	S	GD	IGD	IGD+	HV
mNEAT	Original	0.843	0.992	0.065	0.032	0.621	0.147
		0.297	0.092	0.004	0.006	0.233	0.299
	Archive	0.073	1.144	0.03	0.018	0.036	0.927
		0.2	0.136	0.009	0.004	0.144	0.204
mNEAT-IB	NDR + HV	0.59	1.018	0.083	0.031	0.437	0.408
		0.474	0.08	0.032	0.008	0.352	0.474
	NDR + R2	0.159	1.067	0.065	0.023	0.113	0.84
		0.351	0.11	0.023	0.006	0.261	0.351
	HV	0.477	1.024	0.086	0.029	0.352	0.522
		0.483	0.085	0.033	0.008	0.359	0.484
	R2	0.053	1.145	0.019	0.019	0.036	0.946
		0.211	0.128	0.01	0.004	0.156	0.211

Mean Fitness. The results of the Double Pole Balancing experiment with velocities (Mean Fitness) are shown in Table 3. We examined the commonly used quality indicators ϵ (smallest amount ϵ that is necessary to translate one set A into another set B) [24,25], *Spacing* (short S – the spread of the solutions of a set A), *generational Distance* (short GD – the average distance from a set A to the Pareto front) [25], *Inverted generational Distance* (short IGD – the average distance from the Pareto front to a set A, Pareto Noncompliant), *Inverted generational Distance Plus* (short IGD+ – the average distance from the Pareto front to a set A, weakly Pareto Compliant) [26] and *Hypervolume* [24,25]. See [7, pp. 256–262] for more details on ϵ, S, GD and [26] for IGD and IGD+. The Hypervolume (HV) has been described in Sect. 2.1.

Table 3 shows that the mNEAT-IB (R2) achieves the best results in ϵ, GD, IGD+ (beside mNEAT (Archive)) and HV and second best in IGD. The mNEAT (Archive) performs best with respect to IGD and IGD+ and second best regarding ϵ, GD and HV. To summarize the findings: All examined quality indicators, except Spacing, are dominated by mNEAT-IB (R2) and mNEAT (Archive). Concerning Spacing, the original mNEAT performs best, followed by mNEAT-IB (NDR + HV), while these results are still not good at all: The solutions are concentrated around the most promising areas of the objective space and not spread equidistantly. The mNEAT performs worst regarding all quality indicators, except Spacing (where the mNEAT-IB (R2) gives the worst results) and GD (where the mNEAT-IB (HV) is worst). With respect to the quality indicators shown in Table 3, we find that the mNEAT-IB (NDR + R2), the mNEAT-IB (R2) and the mNEAT (Archive) show statistically significant better results than the mNEAT-IB (NDR + HV), the mNEAT-IB (HV) and the mNEAT in many

cases. This would indicate that this group were to be preferred for further investigations. In contrast, mNEAT shows best results in Spacing, which could be caused by mNEAT's behaviour to replace large parts of the population by new individuals. We assume that mNEAT rather explores the objective space instead of exploiting promising areas. It has to be investigated whether the differences that have been observed between the algorithms and variations concerning the quality indicators depend on the selected parameter settings or the problem instance, therefore further experiments will be carried out in future research.

6 Conclusions and Future Work

This paper focused on multi-objective Neuroevolution. We followed two main research directions both based on the well-known neuroevolutionary approach NEAT which was designed for single-objective tasks. The first focused on developing multi-objective variants of NEAT itself. Here, we introduced a novel indicator-based algorithm. The second direction considered the combination of efficient evolutionary multi-objective algorithms developed for a large number of objectives and NEAT. In this case, the multi-objective algorithms provide the framework into which the main principles of NEAT are integrated. We derived and tested four different variants. This is the first approach which uses these modern multi-objective algorithms with indicator-based selection in the context of Neuroevolution: Previous research utilized the SPEA2 with the original NEAT (Pareto Strength approach is mapping K objectives into a single objective, making it applicable to NEAT without any modifications) or the NSGA-II which is not considered as performant when the number of objectives is relatively large.

All in all this paper is intended as a proof of concept showing that our algorithms are suitable to address multi-objective Neuroevolution. The first experimental results are very promising. Our experimental analysis shows that the novel algorithms are capable of finding an energy efficient cart controller for a multi-objective version of the well-known Double Pole Balancing problem within very few evaluations. Concerning the *Mean Fitness* experiment which addresses the quality of the final Pareto front, we find that all algorithms are capable of evolving good controllers within a predefined number of maximum evaluations. However, statistically significant differences between the algorithms exist. The research will be continued in several directions. First of all, we are currently investigating more difficult variants of the Pole Balancing problem.

Despite the promising first results, further experiments on truly high - dimensional MOPs are necessary to test the algorithms ability of optimizing more than three fitness functions. Currently we are investigating the **F**igh**T**ing **G**ame AI Competition [27], for which we apply our algorithms to create neural networks as controllers for an AI player (with four and five objectives). Preliminary results show that our algorithms are capable of beating already established AI opponents after very few evaluations.

To gain more insights concerning the algorithms' behaviour, especially with respect to robustness, we have to compare a larger number of different configurations in future research. For exploring [8, p. 41f.] the search space of parameter

configurations, we will follow the guidelines provided by the design and analysis of simulation experiments (DASE) [28]. Additionally the newly proposed algorithms have to be compared to other existing algorithms like NEAT-PS or MM-NEAT to assess their performance in comparison to these.

In the future, further areas as for example computer games or maze navigation will be considered in order to investigate the range of applicability of our algorithms. This will provide insights regarding the question whether one and then which of the novel algorithms/variations emerges as preferable in the area of Reinforcement Learning. Additionally, further variations of the algorithms should be investigated to create an even more powerful algorithm for multi-objective Neuroevolution. Another interesting aspect to investigate is in how far the parameters of the algorithms can be automatically configured by the algorithms during execution. Reducing the number of parameters that have to be predefined by the user reduces the complexity of the search space (for parameter configurations) and on the other hand increases the usability of the algorithms. Therefore, it represents an important point of future research.

References

1. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction, vol. 1. MIT press, Cambridge (1998)
2. Stanley, K.O.: Efficient evolution of neural networks through complexification. Ph.D. thesis, Department of Computer Sciences, The University of Texas at Austin (2004)
3. Schrum, J., Miikkulainen, R.: Constructing complex NPC behavior via multi-objective neuroevolution. AIIDE **8**, 108–113 (2008)
4. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
5. Schrum, J., Miikkulainen, R.: Discovering multimodal behavior in Ms. Pac-Man through evolution of modular neural networks. IEEE Trans. Comput. Intell. AI Games **8**(1), 67–81 (2016)
6. van Willigen, W., Haasdijk, E., Kester, L.: Fast, comfortable or economical: evolving platooning strategies with many objectives. In: 16th International IEEE Conference on Intelligent Transportation Systems-(ITSC), 2013, pp. 1448–1455 (2013)
7. Coello, C.A.C., Lamont, G.B., van Veldhuizen, D.A., et al.: Evolutionary Algorithms for Solving Multi-objective Problems, vol. 5. Springer, US (2007). https://doi.org/10.1007/978-0-387-36797-2
8. Eiben, A.E., Smith, J.E.: Introduction to Evolutionary Computing. NCS. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-44874-8
9. Hansen, M.P., Jaszkiewicz, A.: Evaluating the Quality of Approximations to the Non-dominated Set. Department of Mathematical Modelling, Technical Universityof Denmark, IMM (1998)
10. Brockhoff, D., Wagner, T., Trautmann, H.: On the properties of the R2 indicator. In: Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation, pp. 465–472 (2012)
11. Beume, N., Rudolph, G.: Faster S-metric calculation by considering dominated hypervolume as Klee's measure problem. In: Kovalerchuk, B. (ed.) Proceedings of the Second IASTED International Conference on Computational Intelligence, 20–22 November 2006, pp. 233–238. IASTED/ACTA Press, San Francisco (2006)

12. Emmerich, M., Beume, N., Naujoks, B.: An EMO algorithm using the hypervolume measure as selection criterion. In: Coello Coello, C.A., Hernández Aguirre, A., Zitzler, E. (eds.) EMO 2005. LNCS, vol. 3410, pp. 62–76. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-31880-4_5
13. Beume, N., Naujoks, B., Emmerich, M.: SMS-EMOA: multiobjective selection based on dominated hypervolume. Eur. J. Oper. Res. **181**(3), 1653–1669 (2007)
14. Trautmann, H., Wagner, T., Brockhoff, D.: R2-EMOA: focused multiobjective search using R2-indicator-based selection. In: Nicosia, G., Pardalos, P. (eds.) LION 2013. LNCS, vol. 7997, pp. 70–74. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-44973-4_8
15. Gruau, F.: Cellular encoding as a graph grammar. In: IEE Colloquium on Grammatical Inference: Theory, Applications and Alternatives, pp. 17/1–1710 (1993)
16. Moriarty, D.E., Mikkulainen, R.: Efficient reinforcement learning through symbiotic evolution. Mach. Learn. **22**(1–3), 11–32 (1996)
17. Gomez, F.J., Miikkulainen, R.: Solving non-markovian control tasks with neuroevolution. In: IJCAI, vol. 99, pp. 1356–1361 (1999)
18. Diaz-Manriquez, A., Toscano-Pulido, G., Coello, C.A.C., Landa-Becerra, R.: A ranking method based on the R2 indicator for many-objective optimization. In: IEEE Congress on Evolutionary Computation (CEC), 2013, pp. 1523–1530. IEEE, Piscataway (2013)
19. Gamow, G., Cleveland, J.M., Freeman, I.M.: Physics: foundations and frontiers. Am. J. Phys. **29**(1), 60 (1961)
20. Calvo, B., Santafé, G.: Statistical Assessment of the Differences (2018). https://cran.r-project.org/web/packages/scmamp/vignettes/Statistical_assessment_of_the_differences.html
21. Friedman, M.: The use of ranks to avoid the assumption of normality implicit in the analysis of variance. J. Am. Stat. Assoc. **32**(200), 675–701 (1937)
22. García, S., Fernández, A., Luengo, J., Herrera, F.: Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power. Inf. Sci. **180**(10), 2044–2064 (2010)
23. Shaffer, J.P.: Modified sequentially rejective multiple test procedures. J. Am. Stat. Assoc. **81**(395), 826–831 (1986)
24. Fonseca, C.M., Knowles, J.D., Thiele, L., Zitzler, E.: A tutorial on the performance assessment of stochastic multiobjective optimizers. In: Third International Conference on Evolutionary Multi-Criterion Optimization (EMO 2005), vol. 216, p. 240 (2005)
25. Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., Da Fonseca, V.G.: Performance assessment of multiobjective optimizers: an analysis and review. IEEE Trans. Evol. Comput. **7**(2), 117–132 (2003)
26. Ishibuchi, H., Masuda, H., Nojima, Y.: A study on performance evaluation ability of a modified inverted generational distance indicator. In: Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation, pp. 695–702 (2015)
27. Lu, F., Yamamoto, K., Nomura, L.H., Mizuno, S., Lee, Y., Thawonmas, R.: Fighting game artificial intelligence competition platform. In: IEEE 2nd Global Conference on Consumer Electronics (GCCE), 2013, pp. 320–323 (2013)
28. Kleijnen, J.P.C.: Design and Analysis of Simulation Experiments, vol. 20. Springer, US (2008). https://doi.org/10.1007/978-0-387-71813-2