Combining Local and Global Search: A Multi-objective Evolutionary Algorithm for Cartesian Genetic Programming

Paul Kaufmann and Marco Platzner

Abstract This work investigates the effects of the periodization of local and global multi-objective search algorithms. We rely on a model for periodization and define a multi-objective evolutionary algorithm adopting concepts from Evolutionary Strategies and NSGAII. We show that our method excels for the evolution of digital circuits on the Cartesian Genetic Programming model as well as on some standard benchmarks such as the ZDT6, especially when periodized with standard multi-objective genetic algorithms.

1 Introduction

Pareto-based multi-objective genetic algorithms show excellent performance when optimizing for multiple and often conflicting goals. In our work, we are interested in multi-criteria optimization of digital hardware [12, 16] using the Cartesian Genetic Programming model [20] to represent circuits. Experience shows that for this specific application domain global multi-objective genetic optimizers can be rather slow, especially when compared with local Evolutionary Strategy (ES) techniques [17]. However, in the presence of multiple objectives local search techniques typically work with fitness functions that are linear combinations of the single objectives, rather than with the Pareto-based principle. Linear weighting schemes need to be balanced by a human designer each time there is a new set of goal functions, in order to obtain the highest possible performance. The demand for an unsupervised and preference-free multi-objective method for CGP is a challenge we address in the presented work.

We describe a *periodization technique* that alternates the execution of global and local evolutionary optimizers [13]. The technique relies on a *periodized execution*

P. Kaufmann (✉) · M. Platzner
Computer Science Department, Paderborn University,
Warburger Str. 100, 33098 Paderborn, Germany
e-mail: Paul.Kaufmann@gmail.com

M. Platzner
e-mail: Platzner@uni-paderborn.de

model that blends algorithm properties, functionalities and convergence behaviors in a simple and straight-forward way. A typical example is the combination of global search for the early phase of an optimization run with local search for the final phase [7, 25].

Additionally, we present a local search algorithm termed *hybrid Evolutionary Strategy (hES)*, a synthesis of a standard ES and a Pareto set preserving technique, and investigate its performance when periodized with multi-objective genetic optimizers NSGAII [4] and SPEA2 [29]. The characteristic of hES is that it applies a $\mu + \lambda$ ES on the Pareto-dominant individuals obtained by a multi-objective genetic algorithm while keeping diversity in and avoiding deterioration of the Pareto set.

The remainder of this Chapter is structured as follows: Sect. 2 presents related work on hybrid evolutionary search techniques. Our periodization model is defined in Sect. 3, followed by a discussion of the hybrid Evolutionary Strategy (hES) in Sect. 4. Section 5 defines the fitness metrics used in our experiments and Sect. 6 shows the benchmarks and presents the results. Finally, Sect. 7 concludes the work.

2 Related Work

Early work on multi-objective optimization of CGP was presented by Kaufmann and Platzner in [14]. The authors used NSGAII and SPEA2 for the optimization of Boolean circuits for functional quality, area, and delay. In the following years the authors have refined their method in [11, 24]. In the meantime, Walker et al. have also proposed a similar approach for the optimization of digital circuits regarding multiple objective in [27].

Hernández-Díaz et al. [7] presented a two-stage multi-objective evolutionary algorithm based on Differential Evolution (DE) and Rough Sets (RS) theory. In the first stage, the authors employed a fast converging multi-objective DE scheme to compute an initial Pareto frontier approximation. In the second stage, they improve the Pareto set diversity using RS theory for detecting loosely-covered regions. The algorithm's performance is verified on the standard ZDT$\{1, \ldots, 6\}$ and DTLZ $\{1, \ldots, 4\}$ benchmarks [6, 19]. To compare the computed Pareto sets, the authors used three metrics, the unary additive epsilon indicator [32], the standard deviation of crowding distances (SDC) [5] and the space covered by a Pareto set [31]. The proposed algorithm generally outperformed NSGAII, except on the DTLZ2 and DTLZ4 benchmarks using the SDC metric.

Talbi et al. [25] proposed a similar two-stage approach and used a multi-objective genetic algorithm (GA) to calculate a first rough Pareto frontier approximation, followed by a local search technique for refining the approximation. The authors observed improved behavior to a GA-only approach as soon as the complexity of the test problems increases.

Zapotecas et al. [30] presented a hybrid approach combining the global optimizer NSGAII with the local optimizers of Nelder and Mead [21] and the golden section method. The authors enhanced the exploratory NSGAII by local search methods in order to reduce the number of fitness evaluations. The hybrid algorithm was compared to standard NSGAII on continuous benchmarks ZDT$\{1, \dots, 4\}$, ZDT6 and DTLZ$\{1, 2\}$ using the metrics inverted generational distance [26], spacing [22] and coverage indicator [28]. With the exception of the ZDT6 and DTLZ$\{1, 2\}$ benchmarks in combination with the spacing metric, the hybrid algorithm outperformed NSGAII.

Harada et al. [8] analyzed *GA-and-LS* and *GA-then-LS* schemata in which local search is applied either after each generation or after a completed run of a genetic algorithm. The authors concluded that *GA-then-LS* is superior to *GA-and-LS* on multiple benchmarks and used generational and Pareto-optimal frontier distances [5] for comparison.

The work of Ishibuchi et al. [9, 10] is close to our approach. The authors discuss various implementations of standard multi-objective optimizers such as SPEA2 and NSGAII combined with local search. The key idea of their approach is to periodically swap between different optimizers during a run. The authors conclude that the performance of such a hybrid optimizer is sensitive to the balance between global and local search. However, by carefully weighting global and local search strategies the periodized hybrid optimizer outperformed the standard multi-objective optimizer.

In our work, which base on the work of Kaufmann et al. [13], we investigate hybrid Evolutionary Strategies (hES) and its periodization with the multi-objective optimizers NSGAII and SPEA2 in a *GA-and-LS* manner.

3 The Periodization Model

Let $A = (a_1, a_2, \dots, a_n)$ be the set of algorithms used in the periodization. As an illustrative example, consider $A = \{\mathrm{GA1}, \mathrm{GA2}, \mathrm{LS}\}$. For a hypothetical periodized algorithm that executes a single step/generation of GA1, followed by two steps of LS, then a single step of GA2 and two steps of LS, the index sequence I for the algorithm selection is given by (a_1, a_3, a_2, a_3), and the repetition sequence F is $(f_1, f_2, f_3, f_4) = (1, 2, 1, 2)$. While in this specific example, F is a vector of constants, the number of repetitions can be adaptively adjusted based on the history of the optimization run $\mathscr{H}$. In particular, global search GAs with fast convergence in the beginning of an optimization run could be repeated more often in the early search phases, while local search algorithms that excel at improving nearly optimal nondominated sets could be used more intensively in the final optimization phase.

With t as the current generation number, $\mathscr{H}$ as the history of the current optimization run, $A = (a_1, a_2, \dots, a_n), n \in \mathbb{N}$ as the set of algorithms used in the periodization, $I = (i_1, i_2, \dots, i_m), m \in \mathbb{N}, i_k \in (1, 2, \dots, n)$ as the set of indices for the selected algorithms in the execution sequence, and $F = (f_1, f_2, \dots, f_m), f_k(t, \mathscr{H}) \rightarrow \mathbb{N}$ as the number of repetitions for the algorithms in I, the complete *periodized execution model P* is defined as:

$$P := A_I^F = (a_{i_1}^{f_1(t,\mathscr{H})}, a_{i_2}^{f_2(t,\mathscr{H})}, \dots, a_{i_m}^{f_m(t,\mathscr{H})}).$$

The history $\mathscr{H}$ can be large if considering the complete information of an optimization run, or more compact if considering, for example, only the dominated space of the current nondominated set. In our experiments, we choose $f_k(t, \mathscr{H})) := f_k(t) \equiv$ *const*. For the general case, however, $\mathscr{H}$ changes with each generation. Accordingly, the number of algorithm repetitions $f_k(t, \mathscr{H}_t)$ computed in generation t for algorithm a_k may differ from the $f_k(t + 1, \mathscr{H}_{t+1})$ computed in the next generation. Therefore, the repetition vector F needs to be updated after each generation. An example where this effect becomes relevant is when some algorithm is iterated until local convergence occurs. That is, if for l algorithm repetitions the best individual or the nondominated area does not change, the periodization scheme proceeds with the next algorithm. However, if the population can be improved, the algorithm is executed again for at least l generations.

4 Hybrid Evolutionary Strategies

Evolutionary Strategies in their original form rely solely on a mutation operator. The $\{\mu \overset{,}{+} \lambda\}$ ES uses μ parents to create λ offspring individuals and selects μ new parents from all individuals in case of a '+' variant or from the new individuals in case of the ',' variant.

hES is a $1 + \lambda$ ES designed for periodization with multi-objective evolutionary algorithms. In particular, we include two concepts from the Elitist Nondominated Sorting GA II in hES: fast nondominated sorting and crowding distance as a diversity metric. Fast nondominated sorting calculates nondominated sets for the objective space points. The *crowding distance* for a point is defined as the volume of a hypercube bounded by the adjoining points in the same nondominated set. Consequently, the crowding distance creates an order, denoted by $\prec_n$, on the points of a nondominated set. hES uses fast nondominated sorting to decompose parents and offspring individuals into nondominated sets, and uses crowding distances to decide which of the individuals might be skipped in order to keep the nondominated set diverse. In summary, the key ideas are:

1. A local search style algorithm is executed for every element of a given set of solutions. Exactly one individual, which is nondominated, from a parent and its offspring individuals proceeds to the next population.
2. Offspring individuals that are mutually nondominated to their parent but have a different Pareto vector are skipped. This prevents unnecessary fluctuations in the nondominated set.
3. Neutral genetic drift, as presented by Miller in [20], is achieved by skipping a parent if at least one of its offspring individuals holds an equal Pareto vector.
4. Parents and offspring individuals are partitioned into nondominated sets and new parents are selected using NSGAII's crowding distance metric.

Algorithm 1 shows the pseudocode of an hES implementation, `hES-step`. The algorithm starts with the creation of offspring individuals in lines 1–4. To this end, for every individual in the parent population P_t, `hES-step` executes $1 + \lambda$ ES appending the newly created offspring individuals to Q_t. The $1 + \lambda$ ES loop is implemented by the `ES-generate` in Algorithm 1. After the offspring individuals are created, `hES-step` proceeds with the concatenation of parents and offspring individuals by calling the `add-replace` procedure, listed in Algorithm 3. `add-replace` clones the parent population and successively adds offspring individuals that have a unique Pareto vector to this population. An offspring individual with a Pareto vector identical to its parent replaces the parent. Then, `hES-step` partitions the concatenated set R_t in line 6 into nondominated sets $\mathscr{F}_i$ using NSGAII's `fast-nondominated-sort`. After that, starting with the dominant set $\mathscr{F}_1$, the algorithm partitions $\mathscr{F}_1$ by the parents into $G = \{G_1, G_2, \ldots\}$. That means all individuals of G_i have the same parent p. Additionally, if $p \in \mathscr{F}_1$, then $p \in G_i$. Should a non-empty set G_i not contain the parent p, one of the least crowded individuals of G_i is selected to proceed to the next generation. Otherwise, the parent proceeds to the next generation. Once p or one of its offspring individuals is transferred to the next generation, p and all of its offspring individuals are skipped by `hES-step` from further processing in the currect generation.

Algorithm 1: `hES-step`(λ,P_t)—perform a single hES step

```
   Input: $\lambda$, parent population $P_t$
   Output: new archive $P_{t+1}$
1  $Q_t \leftarrow \emptyset$
2  foreach $p \in P_t$ do
3      $Q_t \leftarrow Q_t \cup$ ES-generate($p$, $\lambda$)
4  end
5  $R_t \leftarrow$ add-replace($P_t$, $Q_t$)
6  $\mathscr{F} \leftarrow$ fast-nondominated-sort($R_t$)
7  $P_{t+1} \leftarrow \emptyset$
8  foreach $\mathscr{F}_i \in \mathscr{F}$ do
9      crowding-distance-assignment($\mathscr{F}_i$)
10     $\mathscr{G} \leftarrow$ group-ordered-by-parent($\mathscr{F}_i$)
11     foreach $\mathscr{G}_j \in \mathscr{G}$ do
12         if parent of $\mathscr{G}_j$ not already replaced then
13             if parent($\mathscr{G}_j$) $\in \mathscr{G}_j$ then
14                 $P_{t+1} \leftarrow P_{t+1} \cup$ {parent($\mathscr{G}_j$)}
15             else
16                 sort($\mathscr{G}_j$, $\prec_n$)
17                 $P_{t+1} \leftarrow P_{t+1} \cup \{\mathscr{G}_j[0]\}$
18             end
19             mark parent of $\mathscr{G}_j$ as replaced
20         end
21     end
22 end
```

Algorithm 2: `ES-generate(`p,λ`)`—generate λ offspring individuals

```
    Input: parent $p$, number of offspring individuals $\lambda$
    Output: offspring set $Q$
1   $Q \leftarrow \emptyset$
2   for $i \leftarrow 1$ to $\lambda$ do
3       $p' \leftarrow$ mutate($p$)
4       $Q \leftarrow Q \cup \{p'\}$
5   end
```

Algorithm 3: `add-replace(`P,Q`)`—return copy of P joint by Q, replace parents in P by offspring individuals in Q with equal Pareto vectors, avoid adding multiple offspring individuals with equal Pareto vectors.

```
    Input: sets $P$, $Q$
    Output: set $R$
1   $R \leftarrow P$
2   foreach $q \in Q$ do
3       if $\nexists r \in R : r \preceq q \wedge q \preceq r$ then
4           $R \leftarrow R \cup \{q\}$
5       end
6       if $\exists r \in R : r \preceq q \wedge q \preceq r \wedge parent(\{q\}) == r$ then
7           $R \leftarrow R \cup \{q\}$
8           $R \leftarrow R \backslash \{r\}$
9       end
10  end
```

5 Performance Assessment

To analyze the performance of multi-objective optimizers, we need to compare the calculated Pareto sets. In this work we employ two methods: the ranking of Pareto sets by a quality indicator and the analysis of the mean Pareto set, attained during multiple runs. Both methods are described by Knowles et al. [18] and are also implemented in the PISA toolbox by Bleuler et al. [1].

5.1 Quality Indicators

To compare Pareto sets, Zitzler et al. [32] introduced the concept of a Quality Indicator (QI) as a function mapping a set of Pareto sets to a set of real numbers. Under QI, the Pareto sets define a relation on the Pareto set quality. In our work, we use the unary additive epsilon indicator $I^1_{\varepsilon+}$. It is based on the binary additive epsilon indicator $I_{\varepsilon+}$ which is defined for two Pareto sets A and B as:

$$I_{\varepsilon+}(A,B) = \inf_{\varepsilon \in \mathbb{R}} \{\forall b \in B \; \exists a \in A \,:\, a \preceq_{\varepsilon+} b\}.$$

Table 1 Interpretation of the Kruskal-Wallis test: given the Kruskal-Wallis test rejects H_0, a dot denotes a p-value higher than α

	A_1	A_2	A_3
A_1	–	0.002	0.007
A_2	·	–	·
A_3	·	0.003	–

Here, the relation $\preceq_{\varepsilon+}$ is defined as $a \preceq_{\varepsilon+} b \Leftrightarrow \forall i : a_i \leq \varepsilon + b$. For a reference Pareto set R, the unary additive epsilon indicator $I^1_{\varepsilon+}$ can be now derived as

$$I^1_{\varepsilon+}(A) = I_{\varepsilon+}(A, R).$$

Following Knowles et al. [18], we use the non-parametric Kruskal-Wallis (KW) test [2] to statistically evaluate sequences of quality numbers. The Kruskal-Wallis test differentiates between the null hypothesis H_0 = "The distribution functions of the sequences are identical" and the alternative hypothesis H_A = "At least one sequence tends to yield better observations than another sequence". In case the test rejects H_0, we provide for all sequence pairs the one-tailed p-value. Table 1 presents an example: for an algorithm tuple $(A_{\text{row}}, A_{\text{col}})$ a p-value equal or below α indicates a lower mean for A_{row}. Thus, one can conclude for Table 1 that A_1 outperforms A_2 and A_3, and A_3 outperforms A_2. In our experiments, we configure the significance level α to 0.01.

5.2 *Empirical Attainment Functions*

An additional way of interpreting the results of multi-objective optimizers is to look at the Pareto points that are covered, i.e., weakly dominated, with a certain probability during the multiple repetitions of an optimization algorithm. All Pareto points that have been reached in $x\%$ of the runs are referred to as the $x\%$-attainment. The attainment allows for a direct graphical interpretation as shown in the examples of Figs. 2 and 3.

In order to statistically compare the attainments we use the two-tailed Kolmogorov-Smirnov test [23]. It distinguishes between H_0= "Sequences A and B follow the same distribution" and H_A= "Sequences A and B follow different distributions". Table 2 contains exemplary results for the Kolmogorov-Smirnov (KS) test. It can be interpreted as: A_1 differs significantly from A_2 and A_3. In our experiments, we configure the significance level α to 0.05.

Table 2 Interpretation of the Kolmogorov-Smirnov test: A dot denotes an accepted H_0 hypothesis at the given α

	A_1	A_2	A_3
A_1	–	*	*
A_2	*	–	·
A_3	*	·	–

* indicates significantly different nondominated set distributions

6 Evaluation

We experimented with several benchmarks to compare hES and the periodized variants of hES, NSGAII and SPEA2. At first, we used the standard benchmarks for multi-objective algorithms DTLZ$\{2, 6\}$ and ZDT6. These benchmarks are available with the PISA toolbox [1] and are described in [19]. Second, we compared our algorithms on the evolution of digital circuits, i.e., even 5- and 7-parity and $(2, 2)$ and $(3, 3)$ adders and multipliers, using Cartesian Genetic Programming (CGP) [20] as the hardware representation model. Figure 1 illustrates the CGP phenotype. Besides the functional quality of the digital circuit, which in this case is set as a constraint, we select the circuit's area and speed to define a multi-objective benchmark [15].

In our experiments we executed 20 repetitions for every combination of goal function and algorithm. For the hES, `ES-generate` produces 32 offspring individuals for each parent. For the other benchmarks, Algorithm 1 was configured to have one offspring individual per parent.

Table 3 presents the configuration of the benchmarks DTLZ$\{2, 6\}$ and ZDT6. For these benchmarks, NSGAII and SPEA2 employ the SBX crossover operator [3]. The optimization runs were stopped after 10,000 fitness evaluations. Table 4 shows for the digital circuit benchmarks the CGP configurations, termination criteria, and

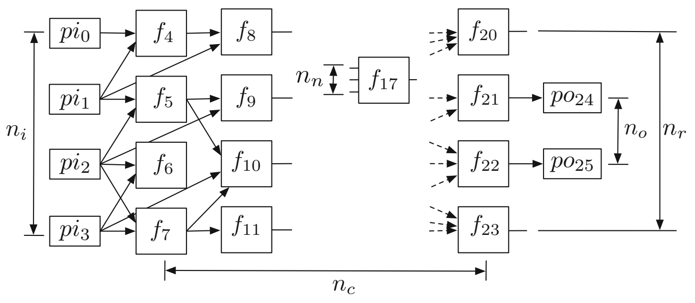


Fig. 1 Cartesian Genetic Programming (CGP) encodes a two dimensional grid of functional units connected by feed forward wires, thus forming a directed acyclic graph. The CGP model is parametrized with the number of primary inputs n_i and outputs n_o, number of rows n_r and columns n_c, number of functional block inputs n_n, the maximal length of a wire l and the functional set F that can be computed by the nodes

Table 3 DTLZ2, DTLZ6, and ZDT6 benchmark configurations

Number objectives	2
Number of decision variables	100
Individual mutation probability	1
Individual recombination probability	1
Variable mutation probability	1
Variable swap probability	0.5
Variable recombination probability	1
Eta mutation	20
Eta recombination	15
Use symmetric recombination	1

Table 4 CGP benchmark configurations. S is the termination number, measured in fitness evaluations. $|P|$ and $|A|$ denote the capacity of the parent population and the archive

	5-parity	7-parity	$(2, 2)$ add	$(2, 2)$ mul	$(3, 3)$ add	$(3, 3)$ mul
S	$400,000$	$800,000$	$400,000$	$1,600,000$	$400,000$	$160,000$
n_i	5	7	4	4	6	6
n_o	1	1	4	4	6	6
n_n	2					
l	∞					
n_c	200					
n_r	1					
$\|P\|/\|A\|$	32/100 for hES, 50/100 else					
F	see Table 5					
P(mut.)	0.1					
P(rcmb.)	0.0 for hES, 0.5 else					
rcmb. type	one-point					

population sizes. We limit the functional set to the Boolean functions presented in Table 5. To simplify the nomenclature, we use the following abbreviations:

$$\begin{aligned} &\text{hES} &&\rightarrow \texttt{h} \\ &\text{SPEA2} &&\rightarrow \texttt{s} \\ &\text{NSGAII} &&\rightarrow \texttt{n} \end{aligned}$$

Table 5 CGP configuration: functional set F

Number	Function
0	0
1	1
2	a
3	b
4	$\overline{a}$
5	$\overline{b}$
6	$a \cdot b$
7	$a \cdot \overline{b}$
8	$\overline{a} \cdot b$
9	$\overline{a} \cdot \overline{b}$
10	$a \oplus b$
11	$a \oplus \overline{b}$
12 13	$a + b$ $a + \overline{b}$
14	$\overline{a} + b$
15	$\overline{a} + \overline{b}$
16	$a \cdot \overline{c} + b \cdot c$
17	$a \cdot \overline{c} + \overline{b} \cdot c$
18	$\overline{a} \cdot \overline{c} + b \cdot c$
19	$\overline{a} \cdot \overline{c} + \overline{b} \cdot c$

6.1 Periodization of hES for DTLZ2, DTLZ6 and ZDT6

To examine the effect of local search, we first execute the standard NSGAII and SPEA2 for a given benchmark in order to determine the reference performance. Then, we increase step by step the influence of local search by periodizing NSGAII with hES until only hES is executed. In terms of our periodization model (cf. Sect. 3), we investigate the six periodization schemes: `n`, `s`, `nh`, `nh`^{4}, `nh`^{10}, and `h`.

Table 6 shows the results of the KW test applied to the benchmarks DTLZ$\{2, 6\}$ and ZDT6 with respect to the unary additive epsilon indicator $I^1_{\varepsilon+}$ at the significance level $\alpha = 1\%$. The results of the KS test are omitted as they indicate differences significant at $\alpha = 5\%$ between the nondominated sets for almost all combinations of algorithm and benchmark.

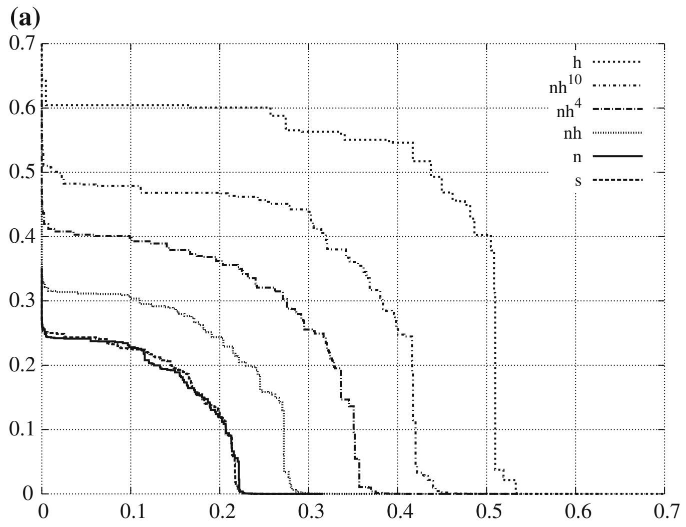
The central observation for the DTLZ2 and DTLZ6 experiments is that the quality of the nondominated sets degrades with an increasing influence of local search. Starting with the periodization of NSGAII and hES, the KW test shows falling performance when increasing the number of hES iterations. The hES-only experiment results in the worst performance of all the algorithms. The KW test results are confirmed by the graphical interpretation of the 75% attained nondominated sets in Fig. 2a, b.

Table 6 Comparison of the nondominated sets of DTLZ2, DTLZ6, and ZDT6. A dot denotes an accepted H_0. When the KW test rejects H_0 for the algorithm pair (a_{row}, a_{col}), a one-tailed p-value lower than $\alpha = 0.01$ indicates that a_{row} evolves significantly better nondominated sets than a_{col} regarding $I^1_{\varepsilon+}$

		n	s	nh	nh^4	nh^{10}	h
KW test DTLZ2	n		·	0.0	0.0	0.0	0.0
	s	·		0.0	0.0	0.0	0.0
	nh	·	·		0.0	0.0	0.0
	nh^4	·	·	·		0.0	0.0
	nh^{10}	·	·	·	·		0.0
	h	·	·	·	·	·	
DTLZ6 KW test	n		·	0.0	0.0	0.0	0.0
	s	·		0.0	0.0	0.0	0.0
	nh	·	·		0.0	0.0	0.0
	nh^4	·	·	·		0.0	0.0
	nh^{10}	·	·	·	·		0.0
	h	·	·	·	·	·	
ZDT6 KW test	n		·	·	·	·	0.0
	s	·		·	·	·	0.0
	nh	0.0001	0.0027		·	0.0001	0.0
	nh^4	0.0	0.0	0.0014		0.0	0.0
	nh^{10}	·	·	·	·		0.0
	h	·	·	·	·	·	

For the ZDT6 benchmark, the influence of hES is not as one-sided as for the DTLZ{2,6} benchmarks. The nh periodization outperforms SPEA2 and NSGAII. Further increase in the influence of hES in the nh^4 periodization lets it dominate all other algorithms. The nh^{10} periodization is on a par with SPEA2 and NSGAII, while the execution of hES alone, as with DTLZ{2, 6}, falls behind. The 75% attainments pictured in Fig. 3 confirm this. Interestingly, despite the large gap between nh^{10} and the group of SPEA2 and NSGAII, the KW test finds no significant differences at $\alpha = 1\%$ between the corresponding indicator sequences. The KS test, similar to the results for the DTLZ{2, 6} benchmarks, reveals significant differences at $\alpha = 5\%$ for all algorithm combinations except the NSGAII and SPEA2 pair.

The DTLZ{2,6} and ZDT6 benchmarks demonstrate the various kinds of impact that local search may have when periodized with global search algorithms. To gain an insight into whether the order of the algorithms in the periodization sequence influences the results, and into how an hES-less periodizations of SPEA2 and NSGAII compares to the regular SPEA2 and NSGAII, we fixate on the ZDT6 benchmark and apply 2- and 3-tuple permutations of the NSGAII, SPEA2, and hES algorithms. All the experiments were repeated 100 times and the execution was stopped after 200 generations.



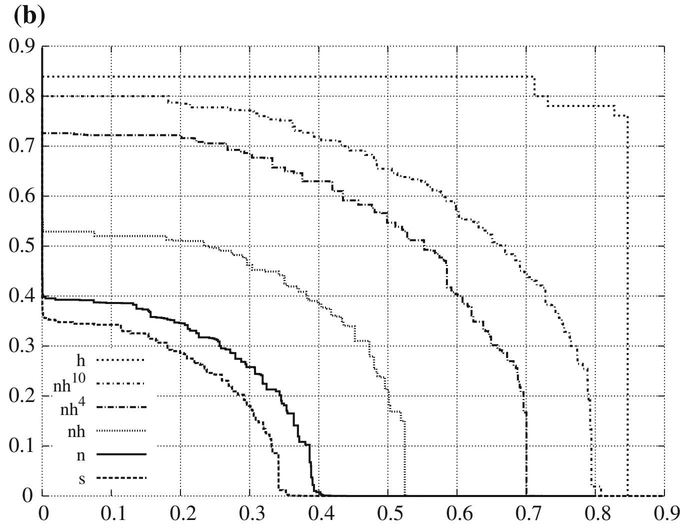


Fig. 2 75%-attainments for the 2-dimensional DTLZ2 **a** and DTLZ6 **b** benchmarks. A hypothetical optimum is located at $\vec{0}$

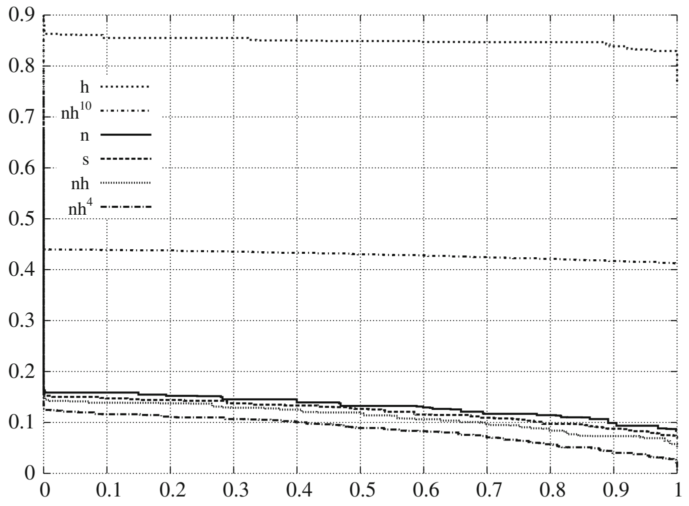


Fig. 3 75%-attainments for the 2-dimensional ZDT6 benchmarks. A hypothetical optimum is located at $\vec{0}$

Table 7 shows the results for 2-tuple combinations of NSGAII, SPEA2, and hES. The general observation taken from the KW test is that hES periodized with either NSGAII or SPEA2 outperforms standard NSGAII, SPEA2, and their combinations. Interestingly, the *hES-after-SPEA2* outperforms the *hES-after-NSGAII*, while *SPEA2-after-hES* does not. This shows that the performance of this particular periodization scheme may be sensitive to the initial order of the executed algorithms.

The KS test confirms the results observed before. There are basically two classes of algorithms, showing significantly different results: the class of algorithms periodized with hES, and the class of NSGAII, SPEA2 and their combinations. In contrast to the previous test, the differences between (hs) and (nh) are now identified as significant.

Next, Table 7 shows also the results of 3-tuple combinations of hES with NSGAII and SPEA2. Similar to the results achieved for the 2-tuple tests, all the periodized algorithms outperform (KW) and differ (KS) from NSGAII and SPEA2.

In summary, we can conclude that for the DTLZ{2, 6} benchmarks, an increasing impact of hES reduces the quality of the nondominated sets evolved, while for the ZDT6 benchmark, the schemes periodized with hES have the dominating results.

6.2 *Periodization of hES for Digital Circuit Design*

In contrast to the previous benchmarks, we optimize for three objectives in this section: the functional quality, the area, and the propagation delay. In total, each

Table 7 ZDT6 nondominated sets comparison for 2- and 3-tuple combinations of NSGAII, SPEA2 and hES to NSGAII and SPEA2. A dot denotes an accepted H_0. When the KW test rejects H_0 for the algorithm pair (a_{row}, a_{col}), a one-tailed p-value lower than $\alpha = 0.01$ indicates that a_{row} evolves significantly better nondominated sets than a_{col} regarding $I^1_{\varepsilon+}$. A star indicates significantly different nondominated set distributions at $\alpha = 0.05$ according to the KS test

		n	s	nh	hn	sh	hs	ns	sn
KW test	n		·	·	·	·	·	·	·
	s	·		·	·	·	·	·	·
	nh	0.0	0.0		·	·	·	0.0	0.0
	hn	0.0	0.0	·		·	·	0.0	0.0
	sh	0.0	0.0	0.0091	·		·	0.0	0.0
	hs	0.0	0.0	·	·	·		0.0	0.0
	ns	·	·	·	·	·	·		·
	sn	·	·	·	·	·	·	·	
KS test	n		·	*	*	*	*	·	·
	s	·		*	*	*	*	·	·
	nh	*	*		·	*	*	*	*
	hn	*	*	·		·	·	*	*
	sh	*	*	*	·		·	*	*
	hs	*	*	*	·	·		*	*
	ns	·	·	*	*	*	*		·
	sn	·	·	*	*	*	*	·	
		n	s	nhs	nsh	shn	snh	hns	hsn
KW test	n		·	·	·	·	·	·	·
	s	·		·	·	·	·	·	·
	nhs	0.0	0.0		·	·	·	·	·
	nsh	0.0	0.0	·		·	·	·	·
	shn	0.0	0.0	·	·		·	·	·
	snh	0.0	0.0	·	·	·		·	·
	hns	0.0	0.0	·	·	·	·		·
	hsn	0.0	0.0	·	·	·	·	·	
KS test	n		·	*	*	*	*	*	*
	s	·		*	*	*	*	*	*
	nhs	*	*		·	·	·	·	·
	nsh	*	*	·		·	·	·	·
	shn	*	*	·	·		·	·	·
	snh	*	*	·	·	·		·	·
	hns	*	*	·	·	·	·		·
	hsn	*	*	·	·	·	·	·	

* indicates significantly different nondominated set distributions

Table 8 The number of circuits with perfect functional quality evolved during 20 runs

	(2, 2) add	(3, 3) add	(2, 2) mul	(3, 3) mul	5-parity	7-parity
n	1	0	0	0	0	0
s	6	0	7	0	0	0
nh	7	0	8	0	0	0
$\mathtt{nh}^4$	9	0	11	0	0	0
$\mathtt{nh}^{10}$	11	0	14	0	0	0
h	12	1	15	0	0	0

algorithm is executed 20 times for each pair of goal functions. Table 8 summarizes the number of runs that resulted in functionally correct solutions. The first observation is that for the parity, the (3, 3) adder, and the (3, 3) multiplier benchmarks, almost none of the algorithms managed to evolve functionally correct circuits. We focus therefore on the experiments involving the (2, 2) adder and the (2, 2) multiplier, when discussing the influence of local search on the evolution of correct circuits.

Despite treating equally all objectives, hES is most effective in finding functionally correct solutions. While SPEA2 outperforms NSGAII for this particular CGP configuration on the (2, 2) adder and multiplier benchmarks, increasing the influence of hES in the periodization with NSGAII produces even greater success rates. $\mathtt{nh}^4$ and especially $\mathtt{nh}^{10}$ periodization schemes reveal only a small gap with the hES-only performance. This insight is also partly confirmed by the results of the KW and the KS tests presented in Table 9. For the (2, 2) adder, the KW test finds no significant differences in nondominated sets at $\alpha = 1\%$ while the KS test partitions the algorithms into groups of {n}, {s, nh, $\mathtt{nh}^4$, $\mathtt{nh}^{10}$}, and {h} with significant differences in the evolved nondominated sets at $\alpha = 5\%$. For the (2, 2) multiplier benchmark, NSGAII is dominated by all, and hES by SPEA2 and nh according to the KW test. The KS test splits the algorithms, similarly to what happened with the (2, 2) benchmark, into groups of {n}, {s, nh, $\mathtt{nh}^4$, $\mathtt{nh}^{10}$}, and {h}. Additionally, in the group of {s, nh, $\mathtt{nh}^4$, $\mathtt{nh}^{10}$} SPEA2 evolves different nondominated sets than it does for $\mathtt{nh}^4$, $\mathtt{nh}^{10}$.

The (3, 3) adder and multiplier benchmarks split the algorithms into {n} and {s, nh, $\mathtt{nh}^4$, $\mathtt{nh}^{10}$, h} groups with different nondominated sets according to the KW test. The KS test reveals, similar to what happened with the (2, 2) benchmarks, the same general tendency of differences between the nondominated sets evolved by the NSGAII, the hES, and the group of the remaining algorithms.

The general partitioning according to the quality of the evolved nondominated sets between NSGAII, hES, and the rest of the algorithms, is even more pronounced for the parity benchmarks, as now the KW test also confirms significant differences (Table 10). That is, SPEA2, nh, $\mathtt{nh}^4$, and $\mathtt{nh}^{10}$ are better than NSGAII and hES for 5- and 7-parity functions and NSGAII is better than hES for the 7-parity function.

Table 9 Non-dominated sets comparison: A dot denotes an accepted H_0. When the KW test rejects H_0 for the algorithm pair (a_{row}, a_{col}), a one-tailed p-value lower than $\alpha = 0.01$ indicates that a_{row} evolves significantly better nondominated sets than a_{col} regarding $I^1_{\varepsilon+}$. A star indicates significantly different nondominated set distributions at $\alpha = 0.05$ according to the KS test

		n	s	nh	nh^4	nh^{10}	h
(2, 2) add KW test	n		·	·	·	·	·
	s	·		·	·	·	·
	nh	·	·		·	·	·
	nh^4	·	·	·		·	·
	nh^{10}	·	·	·	·		·
	h	·	·	·	·	·	
(2, 2) add KS test	n		*	*	*	*	*
	s	*		·	·	·	*
	nh	*	·		·	·	*
	nh^4	*	·	·		·	*
	nh^{10}	*	·	·	·		*
	h	*	*	*	*	*	
(3, 3) add KW test	n		·	·	·	·	·
	s	0.0067		·	·	·	·
	nh	0.0004	·		·	·	·
	nh^4	0.0011	·	·		·	·
	nh^{10}	0.0	·	·	·		·
	h	0.0	·	·	·	·	
(3, 3) add KS test	n		·	*	*	*	*
	s	·		·	·	·	*
	nh	*	·		·	·	*
	nh^4	*	·	·		·	*
	nh^{10}	*	·	·	·		*
	h	*	*	*	*	*	
(2, 2) mul KW test	n		·	·	·	·	·
	s	0.0		·	·	·	0.0
	nh	0.0	·		·	·	0.0013
	nh^4	0.0	·		·		·
	nh^{10}	0.0	·	·			·
	h	0.0016	·	·	·	·	
(2, 2) mul KS test	n		*	*	*	*	*
	s	*		·	*	*	*
	nh	*	·		·	·	*
	nh^4	*	·	·		·	*
	nh^{10}	*	*	·	·		*
	h	*	*	*	*	*	
(3, 3) mul KW test	n		·	·	·	·	·
	s	0.0006		·	·	·	·

(continued)

Table 9 (continued)

		n	s	nh	nh^4	nh^{10}	h
	nh	0.0	·		·	·	·
	nh^4	0.0	·	·		·	·
	nh^{10}	0.0	·	·	·		·
	h	0.0002	·	·	·	·	
(3, 3) mul KS test	n		*	*	*	*	*
	s	*		·	·	*	*
	nh	*	·		·	·	*
	nh^4	*	·	·		·	*
	nh^{10}	*	*	·	·		*
	h	*	*	*	*	*	

* indicates significantly different nondominated set distributions

Table 10 Non-dominated sets comparison: A dot denotes an accepted H_0. When the KW test rejects H_0 for the algorithm pair (a_{row}, a_{col}), a one-tailed p-value lower than $\alpha = 0.01$ indicates that a_{row} evolves significantly better nondominated sets than a_{col} regarding $I^1_{\epsilon+}$. A star indicates significantly different nondominated set distributions at $\alpha = 0.05$ according to the KS test

		n	s	nh	nh^4	nh^{10}	h
5-parity KW test	n		·	·	·	·	·
	s	0.0		·	·	·	0.0
	nh	0.0	·		·	·	0.0
	nh^4	0.0	·	·		·	0.0
	nh^{10}	0.0	·	·	·		0.0
	h	·	·	·	·	·	
5-parity KS test	n		*	*	*	*	*
	s	*		·	·	·	*
	nh	*	·		·	*	*
	nh^4	*	·	·		·	*
	nh^{10}	*	·	*	·		*
	h	*	*	*	*	*	
7-parity KW test	n		·	·	·	·	0.0054
	s	0.0		·	·	·	0.0
	nh	0.002	·		·	·	0.0
	nh^4	0.0	·	·		·	0.0
	nh^{10}	0.0	·	·	·		0.0
	h	·	·	·	·	·	
7-parity KS test	n		*	*	*	*	*
	s	*		·	·	·	*
	nh	*	·		·	·	*
	nh^4	*	·	·		·	*
	nh^{10}	*	·	·	·		*
	h	*	*	*	*	*	

* indicates significantly different nondominated set distributions

The KS test finds significant differences between the three groups, also finding a significant difference between `nh` and `nh`^{10} for 5-parity.

In summary, we can state that periodizations of hES with NSGAII, as well as the non-periodized SPEA2, create, for almost all benchmarks, nondominated sets which are better than those of the non-periodized hES and NSGAII. Additionally, with the increasing influence of hES in a periodization scheme, the probability for the evolution of correct CGP circuits increases.

7 Conclusion

In this work, we investigated the periodization of multi-objective local and global search algorithms. For this, we relied on a periodized execution model and on the hybrid Evolutionary Strategies as a local search technique tailored to periodization with Pareto-based genetic multi-objective optimizers such as NSGAII and SPEA2.

The results show that for the DTLZ$\{2, 6\}$ benchmarks, hES and its periodization with NSGAII underperforms. For ZDT6 and, most importantly, for the evolution of digital circuit benchmarks on the CGP model, hES and its periodizations are significantly better than the reference algorithms NSGAII and SPEA2. Furthermore, the periodized execution model proved to be a simple, fast and flexible approach to combine multiple optimization algorithms for merging functional and behavior properties. Thus, blending multi- and single-objective optimizers, local and global search algorithms and differently converging methods creates a new family of optimization algorithms.

References

1. Bleuler, S., Laumanns, M., Thiele, L., Zitzler, E.: PISA—a platform and programming language independent interface for search algorithms. In: Intlernational Conference on Evolutionary Multi-Criterion Optimization (EMO) LNCS, pp. 494–508. Springer (2003)
2. Conover, W.J., Practical Nonparametric Statistics (3rd edn.). Wiley (1999)
3. Deb, K., Agrawal, R.B.: Simulated binary crossover for continuous search space. Complex Syst. **9**, 115–148 (1995)
4. Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A fast elitist non-dominated sorting genetic algorithm for multi-objective optimisation: NSGA-II. In: Parallel Problem Solving from Nature (PPSN'00), pp. 849–858. Springer (2000)
5. Deb, K., Multi-Objective Optimization Using Evolutionary Algorithms. Wiley, Inc (2001)
6. Deb, K.,Thiele, L., Laumanns, M., Zitzler, E.: Scalable test problems for evolutionary multi-objective optimization. In: Evolutionary Multiobjective Optimization: theoretical Advances and Applications, chap. 6, pp. 105–145. Springer (2005)
7. García, A., Díaz, H., Luis, V., Quintero, S., Carlos, A., Coello, C., Caballero, R., Luque, J.M.: A new proposal for multi-objective optimization using differential evolution and rough sets theory. In: Genetic and Evolutionary Computation (GECCO), pp. 675–682. ACM (2006)

8. Harada, K., Ikeda, K., Kobayashi, S.: Hybridization of genetic algorithm and llocal search in multiobjective function optimization: recommendation of GA then LS. In: Genetic and Evolutionary Computation (GECCO), pp. 667–674. ACM (2006)
9. Ishibuchi, H., Narukawa, K.: Some issues on the implementation of local search in evolutionary multiobjective optimization. In: Genetic and Evolutionary Computation (GECCO), LNCS, pp. 1246–1258. Springer (2004)
10. Ishibuchi, H., Yoshida, T., Murata, T.: Balance between genetic search and local search in hybrid evolutionary multi-criterion optimization Algorithms. In: Genetic and Evolutionary Computation (GECCO), pp. 1301–1308. Morgan Kaufmann Publishers (2002)
11. Paul, K.: Adapting Hardware Systems by Means of Multi-Objective Evolution. Logos Verlag, Berlin (2013)
12. Knieper, T., Defo, B., Kaufmann, P., Platzner, M.: On robust evolution of digital hardware. In: Biologically Inspired Collaborative Computing (BICC), vol. 268 of IFIP International Federation for Information Processing, pp. 2313–222. Springer (2008)
13. Kaufmann, P., Knieper, T., Platzner, M.: A novel hybrid evolutionary strategy and its periodization with multi-objective genetic optimizers. In: IEEE World Congress on Computational Intelligence (WCCI), Congress on Evolutionary Computation (CEC), pp. 541–548. IEEE (2010)
14. Kaufmann, P., Platzner, M.: Multi-objective Intrinsic Hardware Evolution. In: International Conference Military Applications of Programmable Logic Devices (MAPLD) (2006)
15. Kaufmann, P., Platzner, M.: MOVES: a modular framework for hardware evolution. In: IEEE Adaptive Hardware and Systems (AHS), pp. 447–454. IEEE (2007)
16. Kaufmann, P., Platzner, M.: Toward self-adaptive embedded systems: multi-objective hardware evolution. In: Architecture of Computing Systems (ARCS), vol. 4415 of LNCS, pp. 199–208. Springer (2007)
17. Kaufmann, P., Plessl, C., Platzner, M.: EvoCaches: application-specific adaptation of cache mappings. In: IEEE Adaptive Hardware and Systems (AHS), pp. 11–18. IEEE, CS (2009)
18. Knowles, J., Thiele, L., Zitzler, E.: A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers. Technical report, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, Switzerland (2006)
19. Martello, S., Toth, P.: Knapsack Problems: Algorithms and Computer Implementations. Wiley, Inc (1990)
20. Miller, J., Thomson, P.: Cartesian genetic programming. In: European Conference on Genetic Programming (EuroGP), pp. 121–132. Springer (2000)
21. Nelder, J.A., Mead, R.: A simplex method for function minimization. Comput. J. **7**(4), 308–313 (1965)
22. Scott, J.R.: Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization. Master's thesis, Department of Aeronautics and Astronautics. Massachusetts Institute of Technology (1995)
23. Shaw, K.J., Nortcliffe, A.L., Thompson, M., Love, J., Fonseca, C.M.,. Fleming, P.J.: Assessing the performance of multiobjective genetic algorithms for optimization of a batch process scheduling problem. In: Evolutionary Computation, pp. 37–45. IEEE (1999)
24. Lukas, K., Walker, J.A., Kaufmann, P., Plessl, C., Platzner, M.: Evolution of Electronic Circuits. Cartesian Genetic Programming. Natural Computing Series, pp. 125–179. Springer, Berlin (2011)
25. Talbi, El-G., Rahoual, M., Mabed, M.H., Dhaenens, M.C: A hybrid evolutionary approach for multicriteria optimization problems: application to the flow shop. In: International Conference on Evolutionary Multi-Criterion Optimization (EMO), pp. 416–428. Springer (2001)
26. David, A, Veldhuizen, V.: Multiobjective evolutionary algorithms: classifications, analyses, and new innovations. PhD thesis, Department of Electrical and Computer Engineering. Airforce Institute of Technology (1999)
27. Walker, J.A., Hilder, J.A., Tyrrell, A.M: Towards evolving industry-feasible intrinsic variability tolerant cmos designs. In: 2009 IEEE Congress on Evolutionary Computation, pp. 1591–1598 (May 2009)

28. Zitzler, E., Deb, K., Thiele, L.: Comparison of multiobjective evolutionary algorithms: empirical results. In: Evolutionary Computation, vol. 8(2), pp. 173–195. MIT Press (2000)
29. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: improving the strength pareto evolutionary algorithm. Tech. Rep. 103, ETH Zurich (2001)
30. Martínez, S.Z., Carlos, A., Coello, C.: A proposal to hybridize multi-objective evolutionary algorithms with non-gradient mathematical programming techniques. In: Parallel Problem Solving from Nature (PPSN'08), pp 837–846. Springer (2008)
31. Zitzler, E., Thiele, L.: Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. In: IEEE Transcations on Evolutionary Computation, vol. 3(4), pp. 257–271. IEEE 1999
32. 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)