A Hybrid Grammar-based Genetic Programming for Symbolic Regression Problems

Flávio A.A. Motta*, João M. de Freitas*, Felipe R. de Souza*,
Heder S. Bernardino*, Itamar L. de Oliveira*, and Helio J.C. Barbosa*†
*Universidade Federal de Juiz de Fora, Juiz de Fora, MG, Brazil
†Laboratório Nacional de Computação Científica, Petrópolis, RJ, Brazil
flavioaam@hotmail.com, joao@ice.ufjf.br, feliperafael@ice.ufjf.br,
heder@ice.ufjf.br, itamar.leite@ufjf.edu.br, hcbm@lncc.br

***Abstract*—Genetic Programming (GP) is an important technique in evolutionary computing. There has been extensive research and great achievement in GP and its variants. Grammar-based genetic programming (GGP) is one of the most promising ones. We propose here a hybrid approach of GGP with Evolution Strategies (ES). GGP is used to evolve the structure of the models while ES searches for the numerical coefficients in order to improve the overall performance when solving symbolic regression problems. Computational experiments conducted on a set of test-cases reveal that the proposed hybrid approach achieved a good performance when compared to other methods from the literature.**

***Keywords*—Hybrid Technique, Symbolic Regression, Grammar-based Genetic Programming, Evolution Strategy, Differential Evolution.**

I. Introduction

Automating the construction of a computer program is an idea explored by Koza [1] when Genetic Programming (GP) was developed. Based on the natural selection theory, GP is an efficient tool to build computer programs and has been successfully applied in the literature.

Many variants of GP have been developed [2], [3], [4], [5] and Grammar-based Genetic Programming (GGP) is one of the most promising ones [6], [7], [8]. Grammars are largely studied in computer science, as it is a way to enforce type constraints or even define valid expressions [9]. As the search space in GP can be infinite [10], restricting it is a great way to reduce the computational time to find satisfying answers. Also by using formal grammars, GGP can implicitly introduce any preliminary knowledge about the problem. A survey of GP techniques that use formal grammars can be found in [11].

One common problem found when using GGP for symbolic regression is the difficulty in the generation of numerical coefficients [12]. In particular, when the desired model does not involve only integer coefficients, GGP may face difficulties in finding the real-valued coefficients. As all values used by the model are generated by means of the grammar, a specific real number can be generated by a GGP technique when (i) it belongs to the grammar, or (ii) it is a result of arithmetic operations using values and operations allowed by the grammar. It then seems clear that more efficient approaches are desirable and the research for new methods is important.

Using genetic programming together with another technique is very common [13]. Especially when the problem at hand has a particular feature to which GP does not fit very well. This kind of approach is called hybridization, which is the combination of two or more techniques in order to achieve a better performance to solve the problem at hand [14].

Shun and Teo [15] used Differential Evolution (DE) to improve GP's mutation step, and the results are better when using that hybridized method. The algorithm was capable of automatically designing and co-evolving both the controller and the morphology of modular robots.

Funie et al. [16] explored the hybridism of GP with Particle Swarm Optimisation (PSO) for examining trading strategies. In every generation, PSO is used after the fitness selection to enhance the best half of the trading rules found by GP. Tested on 2003 and 2004 market data, those authors concluded that the proposed approach is capable of producing real time feedback in a high frequency trading environment.

Assimi and Jamali [17] applied a Nelder-Mead local search operator to refine the evolution of GP. The proposal was applied to solve the highly nonlinear and nonconvex problem of truss optimization with static and dynamic constraints. It is stated that in the majority of the cases studied, the results obtained by the hybrid algorithm are better than those from other studies reported in the literature.

Alonso et al. [18] studied the application of GP with Evolution Strategy (ES) to symbolic regression problems. After GP evolves its population, ES is used to refine the coefficients found using 25%, 50%, and 75% of the computational effort. Experimental results have shown that the hybrid algorithm outperforms the canonical GP.

Hashimoto et al. [19] state that GP often fails in determining numerical coefficients and proposed a numerical optimization scheme inspired by Particle Swarm. Tests with nonlinear system identification problems were performed and the results obtained by the hybrid approach were more accurate than those found by the standard GP.

Motta et al. [20] also studied the generation of numerical coefficients, and used GP hybridized with DE in order to solve symbolic regression problems. The results showed that the hybrid approach was successful in the majority of the cases.

Evolutionary Strategy (ES) is a popular evolutionary computation technique used to solve optimization problems in $\mathbb{R}^n$.

978-1-5090-6017-7/18/$31.00 ©2018 IEEE

Studies pointed out ES as a good technique to solve continuous optimization problems, when the solution representation is a real-valued vector [21], [22].

We propose here a hybrid method where the GGP technique is combined with ES to solve symbolic regression problems. GGP is used to evolve the structure of the models, while ES searches for the numerical coefficients. As ES was designed to solve optimization problems in continuous search space, its use for finding the real numerical coefficients of the models designed by GGP can improve the quality of the solutions. The proposed technique also updates the grammar by including in it the floating-point values found during the evolution. Symbolic regression problems are used to compare the performance of the technique presented here with a baseline GGP.

II. Grammar-based Genetic Programming

Genetic Programming is a specialized GA, thus also based on the Darwinian natural selection theory [23].

First, GP creates an initial population composed of trees. Each individual represents a solution to the problem. The trees are composed by branches and leaves. In the standard GP these branches may include program operators or mathematical operators, while the leaves may include variables and constants.

During the fitness calculation of each individual created, every model is evaluated based on the objective function. To do so, the algorithm must use the given dataset to calculate how well the model created represents the data available. This step is important because the selection uses the fitness of each individual to determine which ones will be selected to be operated upon by the genetic operators (Crossover and Mutation).

At least two parents are necessary in the crossover step. To generate two offspring, both parents have a node selected and then the sub-trees from these nodes are swapped (the nodes selected must be of the same type). This step is important in the exploration of the search space, since it generates more diversity.

In the mutation step, one parent is selected. In the canonical mutation, one sub-tree of the model is selected. Then a new derivation using the type of the subtree is made. This new derivation now takes place of the selected sub-tree.

The new population is evaluated and then a replacement policy is applied in order to define the individuals that will survive. This process defines a new generation. The algorithm repeats generations until a stop criterion is met. Algorithm 1) presents a pseudo-code for GP.

Grammar-based Genetic Programming is a GP variant. The main difference is that every individual of GGP is produced by means of a grammar. Grammars are composed by terminal and non-terminal symbols. The grammar used defines all the possible syntactic structures of a given individual. In GGP branches are composed by non-terminal symbols, while leaves are composed by terminal symbols. Figure 1a represents the crossover operator in GGP. Figure 1b represents the mutation operator in GGP.

Algorithm 1 Pseudo-code for Genetic Programming.

```
1:  procedure GP
2:      Create initial population pop of size PS
3:      evaluate (pop)
4:      while Termination criteria not met do
5:          selection(pop)
6:          crossover(pop)
7:          mutation(pop)
8:          evaluate(pop)
9:          replacement(pop)
10:     end while
11: end procedure
```

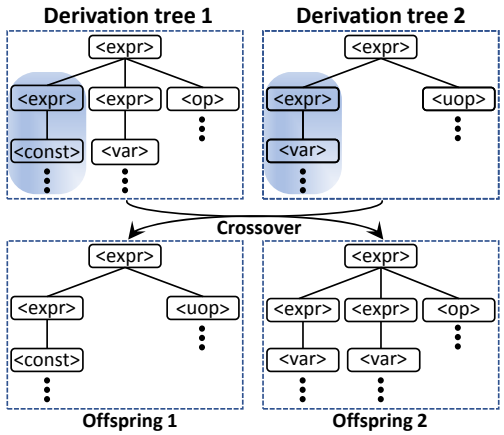
III. Evolution Strategy

Evolution Strategy [24], [25] was developed in the 1970s by Ingo Rechenberg and his co-workers [26]. It uses repeated processes of stochastic variations (mutation) followed by selection, in which new offspring are generated, evaluated and the best ones will be the next generation parents. Differently from most evolutionary techniques, ES uses mainly the mutation operator. This is done by applying a Gaussian mutation. It is important to highlight that ES is commonly applied to solve optimization problems in a continuous search space, making this approach adequate to the context proposed here.

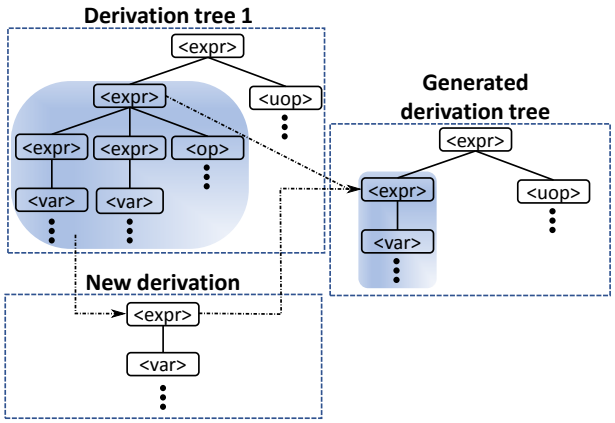
The ES technique requires as parameters the number of parents (μ), the number of offspring (λ), and the replacement strategy. The two main replacement methods in ES are: (i) the $(\mu + \lambda)$-ES, where the new population is composed by the μ best individuals among parents and offspring (see Algorithm 2) while (ii) in the (μ, λ)-ES strategy the μ best offspring are chosen to survive. More information on ES can be found in [27].

Algorithm 2 Pseudo-code for Evolution Strategy

```
1:  procedure ES(μ, λ, ProblemSize)
2:      Pop ← InitializePopulation(μ,ProblemSize)
3:      EvaluatePopulation(Pop)
4:      S_best ← GetBest(Pop,1)
5:      while ¬ StopCondition() do
6:          Children ← ∅
7:          for i ← 0 to λ do
8:              Parent_i ← GetParent(Pop,i)
9:              S_i ← ∅
10:             Si_problem ← Mutate(Pi_problem,Pi_strategy)
11:             Si_strategy ← Mutate(Pi_strategy)
12:             Children ← S_i
13:         end for
14:         EvaluatePopulation(Children)
15:         S_best ← GetBest(Children +S_best,1)
16:         Pop ← SelectBest(Pop,Children,μ)
17:     end while
18:     Return(S_best)
19: end procedure
```



(a) Illustration of the crossover operator.



(b) Illustration of the mutation operator.

Fig. 1. Examples of the crossover and mutation operators of GGP. In (a), the subtrees swapped between the two parents are those not shaded. In (b), the shaded area is the subtree replaced by a new ramdonly generated one.

IV. The Proposed Hybrid Approach

To reach values different from the ones present in the grammar, the model must perform arithmetic operations on them. Those operations may take a lot of space, thus limiting an individual, as GGP individuals have limited depth. Also, smaller trees are simpler to understand, so models represented by smaller trees are preferable.

Here we propose the use of ES to find the numerical coefficients and to improve the grammar during the search. It is important to highlight that ES does not replace any GGP operators. Instead it executes during GGP evolution (after mutation) to refine the numerical coefficients in interleaved intervals. We used the parameter Ω to determine the number of times that ES is executed.

We used the $(1 + \lambda)$-ES strategy. As it is incorporated in GGP we termed it as GGPES. The approach that uses Differential Evolution was named GGPDE. Figure 2 represents a flowchart of the proposed approach during GGP evolution.

We tested three possibilities for the number of individuals that will be optimized by ES: (i) none (standard GGP), (ii) one (the fittest individual), and (iii) an elite (the elite part of the population).

If ES improves the fitness of the model, all numerical coefficients found replace the original ones. Another feature proposed here is that all numerical coefficients found by DE/ES are inserted into the grammar, so that GGP can use them during the standard evolution.

V. Computational Experiments

Some computational experiments are conducted to comparatively evaluate the performance of the proposed hybrid methods. Nicolau et al. [28] provided some guidelines for defining benchmark problems and some functions are presented and discussed in that work. Motta et al. [20] adapted some of the functions presented by Nicolau et al. to contain real-valued coefficients. Also, some new functions are presented in their work. All functions used here are given in Table I. The hybrid GGPs are tested with respect to the symbolic regression of functions with both integer and real-valued coefficients. The results obtained by the proposed approaches are compared to those found by a baseline GGP and by the hybrid GGP presented in [20].

TABLE I. Definition of the functions used in the computational experiments.

Nicolau et al. functions [28]	
F1	$f(x_1, x_2) = 6sin(x_1)cos(x_2)$
F2	$f(x_1, x_2) = (x_1 - 3)(x_2 - 3) + 2sin((x_1 - 4)(x_2 - 4))$
F3	$f(x_1, x_2) = \frac{(x_1-3)^4+(x_2-3)^3-(x_2-3)}{(x_2-2)^4+10}$
F4	$f(x_1, x_2) = \frac{1}{1+{x_1}^{-4}} + \frac{1}{1+{x_2}^{-4}}$
F5	$f(x_1, x_2) = {x_1}^4 - {x_1}^3 + {x_2}^2/2 - x_2$
F6	$f(x_1, x_2) = \frac{8}{2+{x_1}^2+{x_2}^2}$
F7	$f(x_1, x_2) = {x_1}^3/5 + {x_2}^3/2 - x_2 - x_1$
F8	$f(x_1, x_2) = \frac{e^{-(x_1-1)^2}}{1.2+(x_2-2.5)^2}$
Motta et al. functions [20]	
F9	$f(x) = x^4 + 2x^3 - 13x^2 - 14x + 24$
F10	$f(x) = 2x^4 - 9x^3 - 4x^2 + 1$
F11	$f(x) = x^4 + 9x^3 - 3x^2 - x + 1$
F12	$f(x) = x^4 - x^2 + 25x + 2$
F13	$f(x) = 0.5x^4 + 1.5x^3 - 13.5x^2 - 13.5x + 24.5$
F14	$f(x) = 2.5x^4 - 9.5x^3 - 3.5x^2 + 1.5$
F15	$f(x) = 1.5x^4 + 9.5x^3 - 2.5x^2 - 0.5x + 1.5$
F16	$f(x) = 1.5x^4 + 0.5x^3 - 0.5x^2 + 25.5x + 1.5$
F17	$f(x_1, x_2) = 6.7sin(x_1)cos(x_2)$
F18	$f(x_1, x_2) = (x_1 - 3.91)(x_2 - 3.33) + 2.26sin((x_1 - 4.74)(x_2 - 4.53))$
F19	$f(x_1, x_2) = \frac{(x_1-3.96)^4+(x_2-3.41)^3-(x_2-3.11)}{(x_2-2.94)^4+10.57}$
F20	$f(x_1, x_2) = \frac{1.91}{1.93+{x_1}^{-4}} + \frac{1.5}{1.64+{x_2}^{-4}}$
F21	$f(x_1, x_2) = {x_1}^4 - {x_1}^3 + {x_2}^2/2.71 - x_2$
F22	$f(x_1, x_2) = \frac{8.47}{2.59+{x_1}^2+{x_2}^2}$
F23	$f(x_1, x_2) = {x_1}^3/5.92 + {x_2}^3/2.98 - x_2 - x_1$

The context-free grammar used by all the techniques analyzed here is written as follow

<expr>::=<expr><expr><op> | <var> | <expr> <uop> | <const>
<op> ::= + | - | * | / | pow
<uop> ::= exp

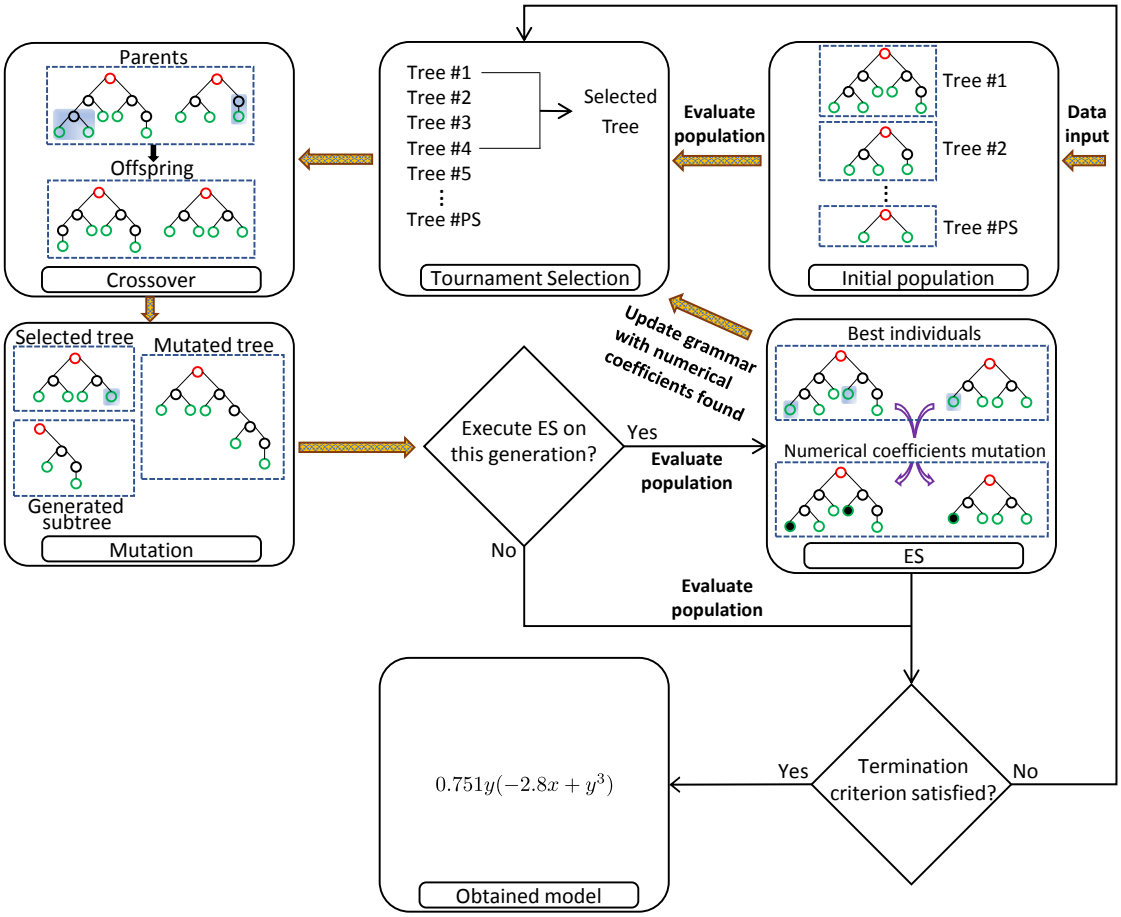


Fig. 2. Flowchart of the proposed approach. The procedure starts from "Data input" while the solution is presented in the "Obtained model" step.

<var> ::= x_1 | x_2
<const> ::= -5 | -4 | -3 | -2 | -1 | 0 | 1 | 2 | 3 | 4 | 5

A maximum number of 100,000 objective function evaluations is used here as the stop condition. For each approach, 30 independent runs were performed for each regression problem.

Also we split the input data into a training set (used to evolve the population), a validation set (used to choose which model fits the best), and a test set, which is used to evaluate the accuracy of the model created. Those sets were randomly sampled and created with 40%, 30%, and 30% of the data, respectively.

In [29] it is suggested the use of Performance Profiles (PPs) to compare the performance of stochastic search techniques when solving constrained optimization problems. Thus, we use PPs of the median values to compare the performance of the techniques proposed.

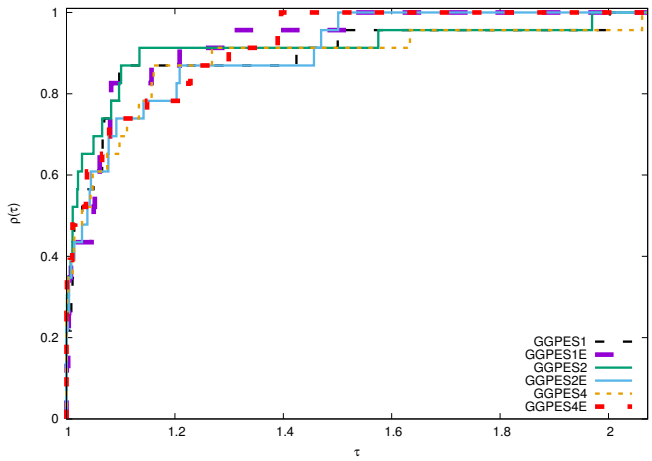
Preliminary experiments were performed to define the number of ES calls (Ω). We choose $\Omega = 10$; so the algorithm will call ES 10 times in equally spaced intervals during the entire process. Also, the best individual generated by ES replaces the original individual from GGP only when an improvement is observed.

No parameter adjustment was made in GGP. We used 8 as the maximum tree depth. The population size used was 500 individuals with an elite of 5%. For selection we used tournament with 2 randomly sampled individuals. Crossover and mutation occur in 90% of the cases.

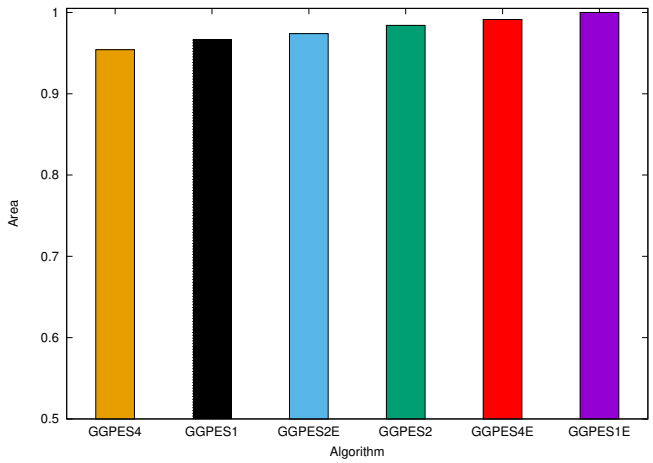
As the PPs may not be convenient to simultaneously analyze a large number of techniques, we performed the analysis of the methods in two stages. First, a parameter analysis of the proposed hybrid GGP and of GGPDE [20] was performed. The best GGPES and the best GGPDE variants are selected according to their Area Under the performance profiles Curve (AUC). Then they are finally compared to a baseline GGP.

Some experiments were initially designed to compare the performance of the techniques with different parameter values. The goal is to select the parameter set with best performance and then compare it with different techniques. A convention for labeling the approaches tested here was created as there are many variants.

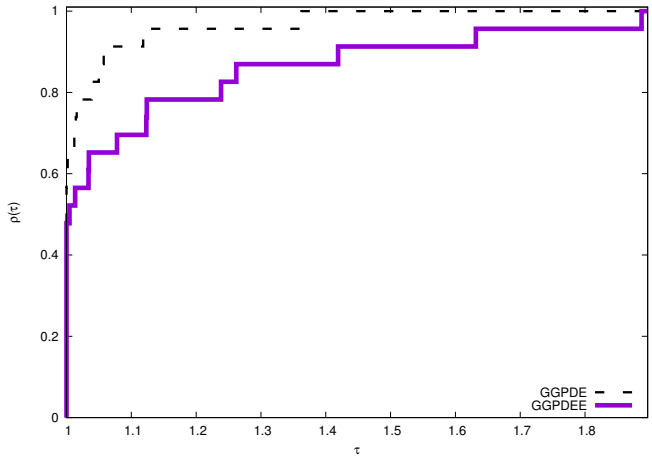
The λ parameter in GGPES was taken from the set $\{1, 2, 4\}$, and the corresponding notation is 1, 2, 4, respectively. The next part of the names given refers to applying the technique only to the best GGP individual (no symbol added to the name) or to the elite part of the population, labeled as 'E'. For example, GGPES4 corresponds to the GGP evolution using (1 + 4)-ES



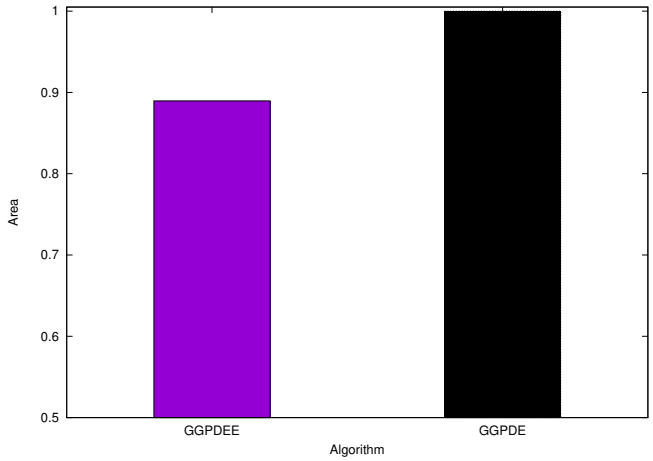
(a) Performance profiles for the results obtained by all approaches using ES. The normalized areas under the performance profiles curves are 0.97, 1.0, 0.98, 0.97, 0.95, and 0.99, respectively, for GGPES1, GGPES1E, GGPES2, GGPES2E, GGPES4, and GGPES4E.



(b) Bar chart of the normalized areas under the performance profiles curves



(c) Performance profiles for the results obtained by all approaches using DE. AUCs are 1.0, 0.89, respectively, for GGPDE, GGPDEE



(d) Bar chart of the normalized areas under the performance profiles curves

Fig. 3. Performance profiles for the results obtained by all hybrid techniques.

applied only to the best individual of the GGP population.

The results for all techniques that use ES during GGP evolution can be seen in Figure 3a. In a general view GGPES1E got better results, as its AUC is the larger one as represented in Figure 3b.

PPs for the results obtained by the GGP techniques with DE are represented in Figure 3c and the normalized areas under the PPs curves are presented in Figure 3d. One can observe that using DE to evolve only the best individual of the GGP population (GGPDE) shows better results.

Figure 4a shows the PPs of the best GGPES, the best GGPDE, and a standard GGP. In Figure 4b, we can see the bar chart for the normalized AUC of the final results.

GGPES1E obtained most of the best results (12 of the 23 regression problems considered here, as shown in Tables II and III). Also it achieved the best median result in the worst case (smaller value such that $\rho(\tau) = 1$), meaning that it is more reliable than the others. GGPES1E achieved the best normalized AUC, 9% larger than the approach with DE, and 5% larger than a baseline GGP.

Tables II and III present the following values for the errors of the models found: minimum, median, mean, standard deviation (Std) and maximum values for each approach applied to all functions tested. Also, the best results are highlighted in boldface.

The proposed approach surpasses the other hybrid algorithm in 19 of the 23 functions used.

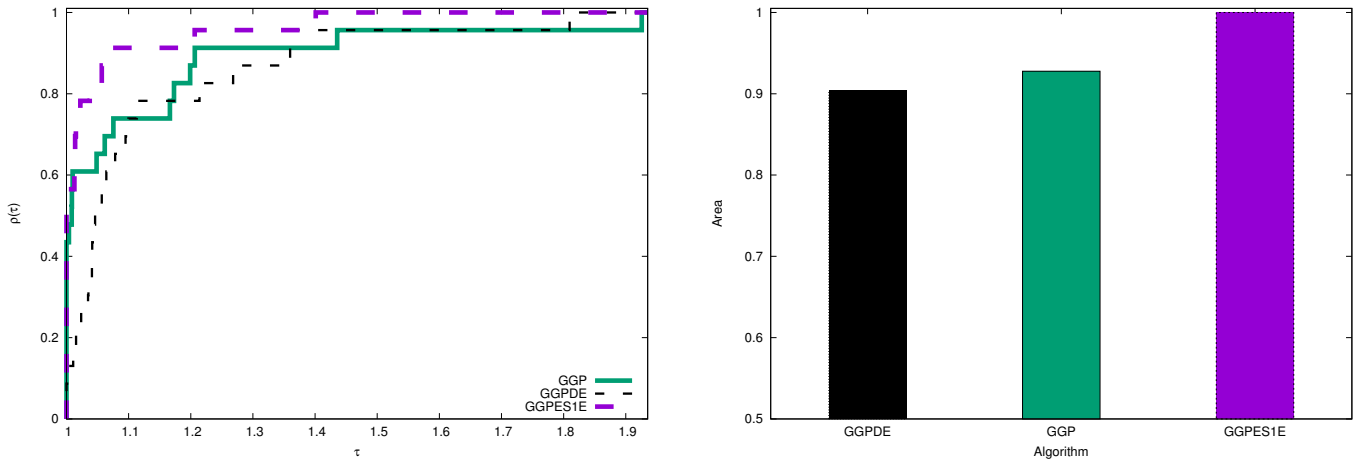
In general, GGPDE achieved smaller AUC and worst results than the baseline GGP. On the other hand, GGPDE presented a higher reliability, due to its better median result in the worst case with respect to GGP. GGPDE overcome the other algorithms in 10 cases when the maximum error is considered.

TABLE II. RESULTS FOR THE MODELS FROM [28].

F	Approach	Min	Median	Mean	Std	Max
F1	GGP	8.55986e+00	9.03084e+00	2.93727e+139	1.58177e+140	8.81181e+140
	GGPDE	**8.31154e+00**	9.00507e+00	9.79067e+35	5.27244e+36	2.93720e+37
	GGPES1E	8.49919e+00	**8.99557e+00**	**1.27945e+01**	**2.05469e+01**	**1.23435e+02**
F2	GGP	1.91048e+00	5.52579e+01	5.78408e+01	3.17618e+01	1.27485e+02
	GGPDE	**1.88257e+01**	5.59574e+01	5.77267e+01	**2.75899e+01**	**1.11872e+02**
	GGPES1E	1.91048e+00	**4.60891e+01**	**5.29214e+01**	3.12387e+01	1.20429e+02
F3	GGP	4.46397e+02	**1.25409e+03**	inf	nan	inf
	GGPDE	7.07300e+02	1.37913e+03	1.44596e+03	**4.71864e+02**	**2.31376e+03**
	GGPES1E	**2.96217e+02**	1.27009e+03	**1.26373e+03**	6.22658e+02	2.48989e+03
F4	GGP	**9.71693e-02**	**1.76233e-01**	**1.76717e-01**	3.37827e-02	**2.28535e-01**
	GGPDE	1.13561e-01	1.80404e-01	1.83878e-01	3.26310e-02	2.47603e-01
	GGPES1E	1.33358e-01	1.87282e-01	1.87338e-01	**3.04711e-02**	2.45933e-01
F5	GGP	6.99766e+02	**2.12322e+03**	**3.53820e+03**	**2.58538e+03**	**8.89310e+03**
	GGPDE	5.51637e+02	2.89011e+03	4.35083e+03	3.24561e+03	9.59374e+03
	GGPES1E	**2.95440e+02**	2.12972e+03	3.81496e+03	2.82688e+03	8.89908e+03
F6	GGP	2.40122e-01	**3.37980e-01**	**3.37526e-01**	3.37993e-02	**3.96340e-01**
	GGPDE*	2.86783e-01	3.59618e-01	3.60859e-01	**2.61792e-02**	4.22752e-01
	GGPES1E	**1.76623e-01**	3.42697e-01	3.44083e-01	3.87147e-02	4.04482e-01
F7	GGP	4.83534e+01	**9.57940e+01**	9.29019e+01	1.65596e+01	1.23517e+02
	GGPDE	4.48136e+01	1.03285e+02	9.80973e+01	**1.54329e+01**	**1.18742e+02**
	GGPES1E	**3.53868e+01**	9.79214e+01	**9.26309e+01**	2.07353e+01	1.30663e+02
F8	GGP	4.75414e-03	1.07161e-02	1.04455e-02	**2.81370e-03**	1.55907e-02
	GGPDE*	**4.63069e-03**	1.16188e-02	1.18554e-02	3.84851e-03	2.30298e-02
	GGPES1E	5.02520e-03	**1.06111e-02**	**9.86379e-03**	3.07961e-03	**1.53554e-02**

TABLE III. RESULTS FOR THE MODELS FROM [20]

F	Approach	Min	Median	Mean	Std	Max
F9	GGP	**1.29678e+02**	7.98313e+02	**7.58367e+02**	1.52837e+02	**9.67035e+02**
	GGPDE	5.92367e+02	7.63638e+02	7.86367e+02	**1.17038e+02**	9.99348e+02
	GGPES1E	5.09145e+02	**7.52078e+02**	7.61722e+02	1.47871e+02	1.31488e+03
F10	GGP	9.35517e+01	3.73518e+03	4.03321e+03	**2.85533e+03**	1.29154e+04
	GGPDE	1.06695e+02	2.45982e+03	3.10349e+03	3.04951e+03	1.43499e+04
	GGPES1E	**1.10938e+01**	**1.93974e+03**	**2.96434e+03**	2.92488e+03	**1.28108e+04**
F11	GGP	**1.08140e+02**	4.11884e+03	6.09788e+03	4.74892e+03	1.94962e+04
	GGPDE	8.15904e+02	3.53245e+03	**4.70760e+03**	**3.65790e+03**	**1.90697e+04**
	GGPES1E	9.99613e+02	**3.41395e+03**	5.13493e+03	5.47124e+03	2.72265e+04
F12	GGP	**4.63924e+00**	3.22735e+02	**4.76098e+02**	5.96483e+02	3.16440e+03
	GGPDE	1.89650e+01	4.97906e+02	5.25856e+02	**3.91861e+02**	**1.56364e+03**
	GGPES1E	3.06325e+01	**2.75155e+02**	4.90343e+02	5.88165e+02	2.65138e+03
F13	GGP	3.53437e+02	6.48406e+02	**6.43104e+02**	1.22059e+02	9.89653e+02
	GGPDE	5.69943e+02	6.69520e+02	6.98879e+02	**1.04421e+02**	**9.77665e+02**
	GGPES1E	**2.75153e+02**	**6.42876e+02**	6.46422e+02	1.60315e+02	9.90008e+02
F14	GGP	**9.98863e+01**	2.69640e+03	4.25789e+03	5.07320e+03	2.22566e+04
	GGPDE	2.11112e+02	2.39309e+03	**2.66089e+03**	**1.62410e+03**	**5.98448e+03**
	GGPES1E	1.09798e+02	**2.31166e+03**	4.18319e+03	7.04165e+03	3.92575e+04
F15	GGP*	1.50551e+03	9.10249e+03	1.12164e+04	7.64453e+03	3.05093e+04
	GGPDE	**2.27566e+02**	6.34047e+03	7.35257e+03	**4.41348e+03**	**2.24148e+04**
	GGPES1E	9.99613e+02	**3.41395e+03**	**5.13493e+03**	5.47124e+03	2.72265e+04
F16	GGP	**6.13112e+02**	**1.49147e+03**	1.86961e+03	1.13160e+03	6.37766e+03
	GGPDE	7.91915e+02	1.66019e+03	**1.78506e+03**	**7.84720e+02**	**3.69971e+03**
	GGPES1E	9.36107e+02	2.08976e+03	2.36118e+03	1.35777e+03	6.39645e+03
F17	GGP	1.01437e+01	**1.11092e+01**	5.55350e+09	2.99065e+10	1.66605e+11
	GGPDE	**1.01427e+01**	1.12288e+01	1.14610e+01	1.60068e+00	1.98315e+01
	GGPES1E	1.02531e+01	1.12599e+01	**1.12503e+01**	**4.33217e-01**	**1.20471e+01**
F18	GGP*	3.46555e+01	7.60644e+01	**8.38878e+01**	**3.76000e+01**	1.90323e+02
	GGPDE*	2.63412e+01	1.02578e+02	8.21005e+02	3.00087e+03	1.59688e+04
	GGPES1E	**1.59719e+01**	**7.54346e+01**	9.08657e+01	4.28061e+01	**1.78512e+02**
F19	GGP	1.43219e+03	3.11123e+03	3.13171e+03	**8.56063e+02**	**5.14068e+03**
	GGPDE	1.65645e+03	**2.89304e+03**	3.25887e+03	1.33690e+03	6.94200e+03
	GGPES1E	**1.13734e+03**	2.93676e+03	**2.99717e+03**	1.17913e+03	6.61260e+03
F20	GGP	**9.46260e-02**	1.56097e-01	1.58346e-01	2.87419e-02	2.37879e-01
	GGPDE	1.05100e-01	1.55742e-01	1.57239e-01	2.54110e-02	2.03227e-01
	GGPES1E	1.05004e-01	**1.48891e-01**	**1.52120e-01**	**2.44020e-02**	**2.01838e-01**
F21	GGP	**5.65314e+01**	**1.98899e+03**	2.24950e+04	1.08169e+05	6.04884e+05
	GGPDE	5.51637e+02	2.89011e+03	4.35083e+03	3.24561e+03	**9.59374e+03**
	GGPES1E	3.58893e+02	2.10183e+03	**3.16602e+03**	**2.44331e+03**	1.00446e+04
F22	GGP	**1.67640e-01**	2.48479e-01	**2.47770e-01**	2.67083e-02	**3.00313e-01**
	GGPDE*	2.23265e-01	2.58630e-01	2.57323e-01	**1.98019e-02**	3.21704e-01
	GGPES1E	2.19848e-01	**2.48478e-01**	2.52500e-01	2.00361e-02	3.24735e-01
F23	GGP	**3.10290e+01**	**4.73336e+01**	**4.60150e+01**	4.10836e+00	5.07818e+01
	GGPDE	3.60356e+01	4.94871e+01	4.77123e+01	5.86100e+00	6.64206e+01
	GGPES1E	3.77568e+01	4.91047e+01	4.73692e+01	**3.47028e+00**	**5.06090e+01**



(a) Best performing techniques; AUCs are, 1.0, 0.91 and 0.95, respectively for GGPES1E, GGPDE and GGP.

(b) Bar chart of the normalized areas under the performance profiles curves

Fig. 4. Performance profiles for the results obtained by GGPDE, GGP, and the proposed GGPES1E.

Also, GGPDE achieved the best standard deviations in 12 (of 23) functions. GGP had the worst performance with respect to standard deviation; best values only in 4 functions. This suggests that GGPDE is more robust than the other ones.

Analyzing the median results, one can observe that GGPES1E obtained the best results in (i) 3 of the 8 functions from Nicolau et al. [28] (7 of them require only integer coefficients), and (ii) 10 of the 15 functions from Motta et al. [20]. Also, GGPES1E had better results in 11 (of 23) functions when mean values are considered.

The Wilcoxon signed-rank test is used here for statistical analysis of the results. In Tables II and III, the asterisks indicate the cases where the results are statistically worse than the best performing technique for that function. It is important to highlight that GGPES1E obtained the best results, or values which are statistically similar to the best ones, in all functions tested here. Thus, one can conclude that GGPES1E outperforms the other techniques in the computational experiments. Also, GGP and GGPDE are marked in two cases each so, in these functions, both approaches can be considered statistically different from the best one.

Finally, it is possible to find "Inf" as a result for F3. That can be explained as the worst model obtained by GGP contains indefiniteness (such as division by zero) when evaluated using the test dataset. This indefiniteness generates a "Not a number (nan)" result in the standard deviation.

VI. Concluding Remarks and Future Works

The hybridization of GGP with ES is proposed here, where ES is used to improve the numerical coefficients of the models obtained by GGP. Also, the proposal improves the formal grammar of GGP during the search, including the numerical coefficients used to generate better models.

The proposed GGPES was compared to other techniques from the literature. Results show that using ES into GGP is useful. This hybrid technique obtained the best results for the problems when evolution during GGP is combined with the $(1 + 1)$-ES strategy and it is applied to the elite part of the population. In particular, the best median results were found by GGPES1E in 12 out of the 23 functions tested.

Other applications can be considered in future work, such as applying the technique to regression problems with more variables, or even to classification problems.

Acknowledgements

The authors thank the reviewers for their comments, and the financial support provided by FAPEMIG (grants APQ-03414-15 and PCE-00439-18), CNPq (grant 310778/2013-1), PPGMC, and PPGCC.

References

[1] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. The MIT Press, 1992.

[2] J. F. Miller and S. L. Smith, "Redundancy and computational efficiency in cartesian genetic programming," *Trans. Evol. Comp*, vol. 10, no. 2, pp. 167–174, Sep. 2006.

[3] T. Perkis, "Stack-based genetic programming," in *Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence*, Jun 1994, pp. 148–153 vol.1.

[4] M. F. Brameier and W. Banzhaf, *Linear Genetic Programming*, 1st ed. Springer Publishing Company, Incorporated, 2010.

[5] S. Harris, T. Bueter, and D. R. Tauritz, "A comparison of genetic programming variants for hyper-heuristics," in *Proc. of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation*. New York, NY, USA: ACM, 2015, pp. 1043–1050.

[6] H. Li and M. L. Wong, "Financial fraud detection by using grammar-based multi-objective genetic programming with ensemble learning," in *2015 IEEE Congress on Evolutionary Computation (CEC)*, May 2015, pp. 1113–1120.

[7] J. M. Luna, M. Pechenizkiy, M. J. del Jesus, and S. Ventura, "Mining context-aware association rules using grammar-based genetic programming," *IEEE Transactions on Cybernetics*, pp. 1–15, 2018.

[8] R. V. Veiga, J. M. de Freitas, H. S. Bernardino, H. J. C. Barbosa, and N. M. Alcântara-Neves, "Using grammar-based genetic programming to determine characteristics of multiple infections and environmental factors in the development of allergies and asthma," in *2015 IEEE Congress on Evolutionary Computation (CEC)*, May 2015, pp. 1604–1611.

[9] R. U. J. Hopcroft, J.E.; Motwani, *Automata Theory, Languages, and Computation, 3rd ed.* Pearson, 2014.

[10] R. Langdon, William B.and Poli, *The Genetic Programming Search Space*. Springer Berlin Heidelberg, 2002, pp. 113–132.

[11] R. I. McKay, N. X. Hoai, P. A. Whigham, Y. Shan, and M. O'Neill, "Grammar-based genetic programming: a survey," *Genetic Programming and Evolvable Machines*, vol. 11, no. 3-4, pp. 365–396, 2010.

[12] M. Evett and T. Fern, "Numeric mutation: Improved search in genetic programming," in *Proc. of the 11th Intl. Florida Artificial Intelligence Research Society Conference*, 1998.

[13] T. A. El-Mihoub, A. A. Hopgood, L. Nolle, and A. Battersby, "Hybrid genetic algorithms: A review," *Engineering Letters*, vol. 13, pp. 124–137, 2006.

[14] C. Grosan and A. Abraham, "Hybrid evolutionary algorithms: Methodologies, architectures, and reviews," in *Hybrid Evolutionary Algorithms*, 2007.

[15] C. W. Shun and J. Teo, "Hybridized gp and self-adaptive de for morphology-controller co-evolution of heterogeneous modular robots," in *Intl. Conf. on Computational Science and Technology*, 2014, pp. 1–6.

[16] A. I. Funie, M. Salmon, and W. Luk, "A hybrid genetic-programming swarm-optimisation approach for examining the nature and stability of high frequency trading strategies," in *2014 13th International Conference on Machine Learning and Applications*, Dec 2014, pp. 29–34.

[17] H. Assimi and A. Jamali, "A hybrid algorithm coupling genetic programming and nelder–mead for topology and size optimization of trusses with static and dynamic constraints," *Expert Systems with Applications*, vol. 95, pp. 127 – 141, 2018.

[18] C. L. Alonso, J. L. Montaña, and C. E. Borges, "Evolution strategies for constants optimization in genetic programming," in *2009 21st IEEE International Conference on Tools with Artificial Intelligence*, Nov 2009, pp. 703–707.

[19] N. Hashimoto, N. Kondo, T. Hatanaka, and K. Uosaki, "Nonlinear system modeling by hybrid genetic programming," *IFAC Proc. Volumes*, vol. 41, no. 2, pp. 4606 – 4611, 2008, 17th IFAC World Congress.

[20] F. A. Motta, H. S. Bernardino, H. J. C. Barbosa, J. M. Freitas, I. L. Oliveira, and F. R. Souza, "A hybrid approach of grammar-based genetic programming and differential evolution for symbolic regression," in *Proc. of the Brazilian Congress on Computational Intelligence*, 2017.

[21] D. Arnold and H.-G. Beyer, "A comparison of evolution strategies with other direct search methods in the presence of noise," *Computational Optimization and Applications*, vol. 24, no. 1, pp. 135–159, 2003.

[22] H. Sharifipour, M. Shakeri, and H. Haghighi, "Structural test data generation using a memetic ant colony optimization based on evolution strategies," *Swarm and Evolutionary Computation*, 2017.

[23] C. Darwin and W. F. Bynum, *The origin of species by means of natural selection: or, the preservation of favored races in the struggle for life*. AL Burt, 2009.

[24] W. Vent, "Rechenberg, ingo, evolutionsstrategie — optimierung technischer systeme nach prinzipien der biologischen evolution," *Feddes Repertorium*, vol. 86, no. 5, pp. 337–337, 1975.

[25] H.-P. P. Schwefel, *Evolution and Optimum Seeking: The Sixth Generation*. New York, NY, USA: John Wiley & Sons, Inc., 1993.

[26] M. Eigen, "Ingo rechenberg evolutionsstrategie optimierung technischer systeme nach prinzipien der biologishen evolution," *mit einem Nachwort von Manfred Eigen*, vol. 45, pp. 46–47, 1973.

[27] H.-G. Beyer and H.-P. Schwefel, "Evolution strategies–a comprehensive introduction," *Natural computing*, vol. 1, no. 1, pp. 3–52, 2002.

[28] M. Nicolau, A. Agapitos, M. O'Neill, and A. Brabazon, "Guidelines for defining benchmark problems in genetic programming," in *2015 IEEE Congress on Evolutionary Computation (CEC)*, 2015, pp. 1152–1159.

[29] H. J. C. Barbosa, H. S. Bernardino, and A. M. S. Barreto, "Using performance profiles to analyze the results of the 2006 CEC constrained optimization competition," in *IEEE Congress on Evolutionary Computation*, 2010, pp. 1–8.