Semantic GP Frameworks: Alignment in the Error Space and Equivalence classes

Stefano Ruberto

GSSI

July 5, 2016
1. Optimization problem
2. Introduction to Genetic Programming
3. Software Engineering point of view
4. A case study
5. Semantic Genetic Programming
6. Overview
7. Semantic space and error space
8. Alignment in the Error Space
9. Reconstruction of the Optimum
10. A New Goal For GP
11. How to search for Optimally Aligned Individuals?
12. ESAGP
13. Experimental Study
14. Computational Complexity
15. ESAGP : observations
16. Future Works: semantic based equivalence classes
Solve an optimization problem to find the best solutions in a (often huge) candidate set.

Considering a pair \((S, f)\) where \(S\) is the set of all possible solutions (search space) and \(f\) the function:

\[ f = S \rightarrow \mathbb{R} \]

\(f\) measures the quality of the solution in \(S\) and is called fitness function.
Global Optimum and Local Optimum

Maximization Problem
We look for the solution \( s \in S \) such that

\[
f(s) \geq f(i), \forall i \in S
\]

Local maxima
the solution \( s \in S \) such that

\[
f(s) \geq f(i), \forall i \in N_s
\]

where \( N_s \) is in the neighbourhood of \( s \), given some criteria
In a regression problem we look for the function $g$ such that $\forall i = 1, 2, \cdots, m$ holds that

$$g(x_{i1}, x_{i2}, \cdots, x_{in}) = y_i$$
The **Genetic Programming** is framed within the broader family of **evolutionary algorithms** (EA) [Bac96]. The EA are inspired by Darwin’s theory of evolution in its various aspects and, specifically for GP, on the **iterative** process based on **reproduction**, **mutation**, **competition** and **selection**.

**Note**

One of the main features of the GP is to output, as a result of the evolution, a real algorithm. **White box** approach as opposed to black box one.
Genetic programming has produced results that can be called “human competitive” from a wide variety of fields, here is some example of successful application [PK14],[ES15]:

- Regression or Classification of non-linear problem.
- In telecommunications: speech quality estimation
- In finance: evolving effective bidding strategies.
- Networks: network coding.
- Clinical applications: Cancer detectors, seizure detectors, mental health diagnosis, etc.
Search Based Software Engineering

Software engineering is ideal for the application of metaheuristic search techniques, such as genetic algorithms, simulated annealing and tabu search. Such search-based techniques could provide solutions to the difficult problems of balancing competing constraints and may suggest ways of finding acceptable solutions in situations where perfect solutions are either theoretically impossible or practically infeasible.

Mark Harman 2001 [HJ01]

- There is usually a need to balance competing constraints.
- Occasionally there is a need to cope with inconsistency.
- There are often many potential solutions.
- There is typically no perfect answer... but good ones can be recognised.
- There are sometimes no precise rules for computing the best solution.
Introduction to Genetic Programming.
Application example in Software Engineering

Fundamental steps

- Representation of the problem.
- Definition of the fitness function

Existing Applications of Optimization Techniques to Software engineering [Har07]

- Accurate cost estimates
- Staff allocations in project planning
- Requirements to form the next release
- Optimizing design decisions
- Optimizing source code
- Optimizing test data generation
- Optimizing test data selection
- Optimizing maintenance and reverse engineering
- ... many others...
Genetic Improvement for Adaptive Software Engineering [HJL+14]
A case study
Performance measure

Using Machine Learning Techniques for Predicting Performance Robustness of Software Under Uncertainty

Measurement-based performance evaluation process in order to support stakeholders in the evaluation of systems that have to meet performance requirements.

- **Uncertainty is critical** in the performance domain when it relates to workload, operational profile, and resource demand.
- It is necessary to **sample uncertain parameters**.
- The application-level monitoring of software system samples is very expensive in terms of **time consumption and resource usage**.
- **Sampled Input Param.**: Number Of Users, Think Time, Number Of Items To Cart, Cpu Catalog.
- **Measured Param.**: Response time, Throughput, CPU Utilization.
A case study

Procedure

Use both model predictions and software measurements to evaluate software performances.
A case study

Results

Results on the **real-world enterprise application**, i.e., the JPetStore

<table>
<thead>
<tr>
<th>Performance Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>$TH$</td>
</tr>
<tr>
<td>Measurements and Predictions</td>
</tr>
<tr>
<td>Measurements</td>
</tr>
</tbody>
</table>

![Graph showing computational time for measurements and predictions](image_url)

<table>
<thead>
<tr>
<th>Computational time (hours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TH</td>
</tr>
<tr>
<td>Meas and Pred</td>
</tr>
<tr>
<td>Measurements</td>
</tr>
</tbody>
</table>
Introduction to Genetic Programming

Solution structure

Unlike other evolutionary algorithms Genetic Programming (GP) has a variable solution size: often tree structures are used.

Functional symbols:

\[ F = \{ f_1, f_2, \ldots, f_n \} \]

Terminal symbols:

\[ T = \{ t_1, t_2, \ldots, t_n \} \]

Example:

\[ y = 2 \times x^2 + x \]

Figure: Syntactic tree.
Introduction to Genetic Programming

Evolution cycle

- **Initialization**: full, grow, RH&H, etc.
- **Selection**: Tournament Selection, Fitness Proportional Selection, Ranking Selection, etc.
- **Genetic operators**: Mutation, Crossover, etc.
- **Fitness function**: RMSE, etc.
**Bloat:** During the evolution, the number of nodes in the trees start growing in a way non proportional to fitness improvement.

**Premature convergence:** The loss of genetic diversity in the population trapped in a local optima.

**Overfitting:** Given a hypothesis space $H$, a hypothesis $h \in H$ is said to overfit the training data if there exists some alternative hypothesis $h' \in H$, such that $h$ has smaller error than $h'$ over the training examples, and $h'$ has a smaller error than $h$ over the entire distribution of instances.
**Goal**: maximize **phenotype** diversity in population to counteract premature convergence exploring larger areas of the **solution space**.

**Structural diversity (**Genotype**) it’s not enough!**

\[ y = 2 \times x^2 + x \]
Taking the test cases input vector $\vec{x}$ the software gives as output the semantic vector $\vec{s}$:

$$
\begin{pmatrix}
  x_{11} & x_{12} & \cdots & x_{1n} & y_1 \\
  x_{21} & x_{22} & \cdots & x_{2n} & y_2 \\
  \vdots & \vdots & \ddots & \vdots & \vdots \\
  x_{m1} & x_{m2} & \cdots & x_{mn} & y_m
\end{pmatrix}
$$

$$\vec{s} \equiv [p(x_{11}, x_{12}, \cdots, x_{1n}), p(x_{21}, x_{22}, \cdots, x_{2n}), \cdots, p(x_{m1}, x_{m2}, \cdots, x_{mn})]$$

The target is expressed as the vector:

$$\vec{t} \equiv [y_1, y_2, \cdots, y_m]$$
Presenting a new way of exploiting semantic awareness and geometry in GP (looking for alignments in the error space ESAGP)

Presenting two different methods that implement this idea

Discussing the obtained results both on training and on test data for real life applications and benchmark problems.

Future works and discussion:
- A general GP framework based on Equivalence Classes EC
- EC incorporate: ESAGP, Linear Scaling and other simple yet effective algorithm.
- Preliminary results.
The target is also represented by a point in the semantic space and usually it does not correspond to the origin.
error vector of an individual $P$ the vector:

$$\vec{e}_P = \vec{s}_P - \vec{t}$$

This is a point in a(nother) $n$-dimensional space, that we call error space.
Each point in the semantic space is translated by subtracting the target. What we get is the error space. In the error space, the target is represented by the origin.
Definition: **Optimally Aligned Individuals**

Two GP individuals $A$ and $B$ are optimally aligned if a scalar constant $k$ exists such that: $\vec{e}_A = k \cdot \vec{e}_B$

- In other words, two individuals are optimally aligned if the straight line that joins their error vectors also intersects the origin.
- If we find two optimally aligned individuals, we are able to reconstruct a globally optimal solution analytically.
Reconstruction of the Optimum

Let A and B be optimally aligned individuals then:

\[ \vec{e}_A = k \cdot \vec{e}_B \]

applying the def. of error vector

\[ \vec{s}_A - \vec{t} = k \cdot (\vec{s}_B - \vec{t}) \]

Obtaining:

\[ \vec{t} = \frac{1}{1-k} \cdot \vec{s}_A - \frac{k}{1-k} \cdot \vec{s}_B \]

Now, we construct an individual with the following genotype:

---

Important!!

The semantics of this individual is equal to \( \vec{t} \) and thus it is a global optimum!
A New Goal For GP

- If we find **two optimally aligned individuals**, then we are able to reconstruct a globally optimal solution analytically.

**Note**
This holds regardless of the quality of each one of these two individuals!

- Thus now the **objective of GP** can be **finding two optimally aligned individuals** (instead of searching directly for a globally optimal solution).
How to search for Optimally Aligned Individuals?

There is a very large number of ways of doing it!

We just implemented one of them.

The idea:

Assume that these are the error vectors of the individuals in a GP population (for instance, the initial population of a GP run)
How to search for Optimally Aligned Individuals?

We consider a particular direction (point) that (informally speaking) “stands in the middle” of the error vectors of the individuals in the population.

We call this point **attractor**.
Than, we make GP “push” the individuals in the population towards alignment with the attractor.
The idea of Optimal Alignment can be extended

**Definition. Optimally Coplanar Individuals**

Three GP individuals A, B and C are optimally coplanar if the bi-dimensional plane on which \( \vec{e}_A, \vec{e}_B \) and \( \vec{e}_C \) lie also intersects the origin of the error space.

- The proof is not given here (see the paper!). It is based on the idea that the concept of optimal coplanarity can be seen as an “iteration” of the concept of optimal alignment. In fact, in this figure and are aligned with \( \vec{e}_A \), and and are aligned with the origin.
Proposed GP Systems

- **ESAGP-1** Objective: finding two individuals whose error vectors are aligned on (i.e. belong to) a straight line that intersects the origin.

- **ESAGP-2** Objective: finding three individuals whose error vectors belong to a bi-dimensional plane intersecting the origin.

- **ESAGP-\(\mu\)** Objective: finding \(\mu + 1\) individuals whose error vectors belong to a \(\mu\)-dimensional plane intersecting the origin.

[RVCS14]
ESAGP-1 maintains an archive of all the "semantically new" individuals that have been found during the GP run.

Every time a new individual $P$ is generated, the algorithm checks whether it is optimally aligned with any of the individuals already in the archive.

If so, the algorithm terminates.

Otherwise, $P$ is added to the archive, unless the archive already contains an individual with the same semantics, and the algorithm continues.
The fitness of an individual is the angle between its error vector and the attractor, and it has to be minimized.

**Error is never used directly** as fitness!

In this work, as attractor, we used:

\[ \vec{a} = \sum_{P \in Pop} \frac{\vec{e}_P}{\|\vec{e}_P\|} \]

Remark: the angle between two vectors \( \vec{e}_A \) and \( \vec{a} \) is:

\[ \theta = \arccos \left( \frac{\vec{e}_A \times \vec{a}}{\|\vec{e}_A\| \cdot \|\vec{e}_A\|} \right) \]
Two complex real-life applications (prediction of pharmacokinetic parameters in drug discovery)

<table>
<thead>
<tr>
<th>Dataset Name(ID)</th>
<th>Number of features</th>
<th>Number of instances</th>
<th>Short goal explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>%F</td>
<td>241</td>
<td>359</td>
<td>Predicting the value of human oral bioavailability of a candidate new drug as a function of its molecular descriptors</td>
</tr>
<tr>
<td>LD50</td>
<td>626</td>
<td>234</td>
<td>Predicting the value of the toxicity of a candidate new drug as a function of its molecular descriptors</td>
</tr>
</tbody>
</table>

Both these datasets are freely available
Experimental Results (Errors)

- ESAGP−1
- ESAGP−2
- GS−GP
- ST−GP

Number of Generations

Training Error (%F)

Test Error (%F)

Training Error (LD50)

Test Error (LD50)

Stefano Ruberto (GSSI)
Semantic GP
July 5, 2016
Experimental Results (Size)

- ESAGP-1
- ESAGP-2
- GS-GP
- ST-GP

Number of Generations

Number of Nodes (%F)

Number of Nodes (LD50)
ESAGP-1 system add no further fitness evaluation to the complexity of standard GP; it just adds comparisons among (error) vectors that have already been calculated (and stored), in order to check for alignment.

In the worst case, ESAGP-1 does \( n \) vector comparisons (where \( n \) is the archive size).

In the worst case, \( n \) is equal to \((\text{population size} \times \text{number of generations})\) (it does not take into account semantic repetitions that are very frequent).

In practice: the overhead of time given by the vector comparisons is compensated by the fact that the ESAGP system evolves smaller trees, so the ESAGP system has approximately the same running time as ST-GP.
Observations

ESAGP

- ESAGP-1 and -2 find comparable solutions to the ones of GS-GP after 350 (even 2000!) generations and better than the ones of ST-GP on the test set in 350 generations.
- **Individuals** evolved by ESAGP-1 are **smaller** than the ones of ESAGP-2, which are smaller than the ones of ST-GP, which are much smaller than the ones of GS-GP
- Adding dimensions seems beneficial (ESAGP-2 consistently outperforms ESAGP-1)
- Looking for **alignment** can be easier than directly looking for a global optimum.

In our experimental study, we have obtained solutions of **comparable quality** as other systems, and **smaller, earlier**.
Future Works: Semantic based equivalence classes

Motivations

- Semantic based **equivalence classes** (EQC)
- Different definitions of equivalence are used
- Exploring the solution space by classes may be **more efficient**
  - One single individual represent the class (**no ”duplicate” solutions**)
  - It is easy to analytically obtain the **best individual of the class**
  - Probably not all the **interesting solutions are** in a solution space **reachable** directly by evolution structured like GP
  - **Interesting equivalence classes** can capture property of the domain (e.g. position invariance, scale invariance, rotation invariance etc.)
E QC concept is actualized by means of two different novel genetic programming systems, in which two different definitions of equivalence are used.

**Equivalent by Translation:** **GPPLUS**

2 individuals are equivalent if $\vec{P}_1 - \vec{P}_2 = \vec{k}$ and $k_i = k_j \quad \forall i, j$

**Equivalent by Scale:** **GPMUL**

2 individuals are equivalent if $\vec{P}_1 / \vec{P}_2 = \vec{k}$ and $k_i = k_j \quad \forall i, j$
Approximate EQC and fitness function

**GOAL**

The goal of these GP system is to evolve $\vec{P}$ equivalent to $\vec{t}$

To measure the distance from the EQC of the target we use the dispersion of values in $\vec{k}$ e.g. the variance

**Filter redundant solutions**

Establishing a threshold on variance of $\vec{k}$ we can reject ”duplicate” semantics.
## Preliminary experimental validation

test problems

<table>
<thead>
<tr>
<th>Dataset</th>
<th># Features</th>
<th># Instances</th>
</tr>
</thead>
<tbody>
<tr>
<td>Airfoil Self-Noise [BPM89]</td>
<td>5</td>
<td>1502</td>
</tr>
<tr>
<td>Concrete Compressive Strength [CVS13]</td>
<td>8</td>
<td>1029</td>
</tr>
<tr>
<td>Parkinson Voice Recording (TOTAL) [CVS14]</td>
<td>19</td>
<td>5875</td>
</tr>
<tr>
<td>Parkinson Voice Recording (MOTOR) [CVS14]</td>
<td>19</td>
<td>5875</td>
</tr>
<tr>
<td>Concrete Slump Test [Yeh09]</td>
<td>9</td>
<td>102</td>
</tr>
<tr>
<td>Yacht Hydrodynamics [OLG07]</td>
<td>6</td>
<td>307</td>
</tr>
</tbody>
</table>
Preliminary experimental validation
filter vs NON-filter

Test set of dataset "Concrete Compressive Strength" filter vs NON-filter

![Graph showing Computational Effort vs Median Best Fitness for FGPPLUS and GPPLUS](image1)

![Graph showing Computational Effort vs Median Best Fitness for FGPMUL and GPMUL](image2)
Performance of GPPLUS and GPMUL are comparable with Linear scaling and Geometric Semantic GP (example on dataset "slump")
## Preliminary experimental validation

### Effect on solutions sizes

<table>
<thead>
<tr>
<th>Technique</th>
<th>Dataset</th>
<th>P-value</th>
<th>% Diff pop. Average size</th>
<th>P-value Best size</th>
<th>% Diff on best ind. Size</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GPMUL</strong></td>
<td>airfoil</td>
<td>0,00</td>
<td>-47,66</td>
<td>0,01</td>
<td>-43,03</td>
</tr>
<tr>
<td></td>
<td>concrete</td>
<td>0,59</td>
<td>10,38</td>
<td>0,53</td>
<td>8,63</td>
</tr>
<tr>
<td></td>
<td>motor</td>
<td>0,34</td>
<td>33,98</td>
<td>0,48</td>
<td>45,32</td>
</tr>
<tr>
<td></td>
<td>motor_total</td>
<td>0,00</td>
<td>-63,76</td>
<td>0,00</td>
<td>-63,00</td>
</tr>
<tr>
<td></td>
<td>slump</td>
<td>0,00</td>
<td>-93,31</td>
<td>0,00</td>
<td>-88,70</td>
</tr>
<tr>
<td></td>
<td>yacht</td>
<td>0,00</td>
<td>-44,85</td>
<td>0,00</td>
<td>-40,39</td>
</tr>
<tr>
<td><strong>GPPLUS</strong></td>
<td>airfoil</td>
<td>0,00</td>
<td>-24,39</td>
<td>0,00</td>
<td>-28,24</td>
</tr>
<tr>
<td></td>
<td>concrete</td>
<td>0,27</td>
<td>-5,77</td>
<td>0,94</td>
<td>0,90</td>
</tr>
<tr>
<td></td>
<td>motor</td>
<td>0,06</td>
<td>-18,88</td>
<td>0,05</td>
<td>-35,05</td>
</tr>
<tr>
<td></td>
<td>motor_total</td>
<td>0,74</td>
<td>-17,98</td>
<td>0,99</td>
<td>-15,07</td>
</tr>
<tr>
<td></td>
<td>slump</td>
<td>0,00</td>
<td>-58,43</td>
<td>0,00</td>
<td>-50,16</td>
</tr>
<tr>
<td></td>
<td>yacht</td>
<td>0,00</td>
<td>-61,48</td>
<td>0,00</td>
<td>-57,62</td>
</tr>
<tr>
<td><strong>LS</strong></td>
<td>airfoil</td>
<td>0,00</td>
<td>-37,70</td>
<td>0,05</td>
<td>-19,02</td>
</tr>
<tr>
<td></td>
<td>concrete</td>
<td>0,19</td>
<td>-6,43</td>
<td>0,33</td>
<td>-10,75</td>
</tr>
<tr>
<td></td>
<td>motor</td>
<td>0,33</td>
<td>-13,22</td>
<td>0,23</td>
<td>6,14</td>
</tr>
<tr>
<td></td>
<td>motor_total</td>
<td>0,23</td>
<td>-26,27</td>
<td>0,27</td>
<td>-27,34</td>
</tr>
<tr>
<td></td>
<td>slump</td>
<td>0,00</td>
<td>-81,94</td>
<td>0,00</td>
<td>-81,09</td>
</tr>
<tr>
<td></td>
<td>yacht</td>
<td>0,00</td>
<td>-63,96</td>
<td>0,00</td>
<td>-63,85</td>
</tr>
</tbody>
</table>

Table 3: For each technique the table shows the percentage variation related to the size of the best individual (last column) and to the average size of the population (third column), at the last generation, when the filter is used. The table also reports the p-values calculated using the Wilcoxon rank-sum test for pairwise data comparison ($\alpha = 0.05$). Bold numbers denote significant results and negative values means and advantage in using filters having a size reduction. For the filtered counterparts, we selected the filter parameter that produces the best training (RMSE) performance. Statistics refer to the same experiments reported in Figures 1 - 12 and Table 2.
Conclusion

- ESAGP and EQC methods do **not optimise directly the error**
- results have **comparable or better** error
- solution size is **smaller**
- ESAGP is very **fast**
- Simple EQC can match **partial input** (advantages over linear scaling)
[Bac96] T. Back. 
Oxford University Press, 1996.

Airfoil self-noise and prediction. 

Prediction of high performance concrete strength using genetic programming with geometric semantic genetic operators. 

Prediction of the unified Parkinson’s disease rating scale assessment using a genetic programming system with geometric semantic genetic operators. 

From evolutionary computation to the evolution of things. 

[Har07] Mark Harman. 
The current state and future of search based software engineering. 

[HJ01] Mark Harman and Bryan F Jones. 
Search-based software engineering. 

Genetic improvement for adaptive software engineering (keynote). 
Geometric semantic genetic programming.  

Examining Semantic Diversity and Semantic Locality of Operators in Genetic Programming.  

[OLG07] I. Ortigosa, R. López, and J. García.  
A neural networks approach to residuary resistance of sailing yachts prediction.  

Genetic Programming.  

[RVCS14] Stefano Ruberto, Leonardo Vanneschi, Mauro Castelli, and Sara Silva.  
ESAGP - a semantic gp framework based on alignment in the error space.  

[Yeh09] I-C Yeh.  
The End