Predict the Success or Failure of an Evolutionary Algorithm

Gopinath Chennupati, Conor Ryan, R. Muhammad Atif Azad
Gopinath.Chennupati@ul.ie, Conor.Ryan@ul.ie, Atif.Azad@ul.ie

Introduction
Grammatical Evolution (GE) [3] is an evolutionary algorithm (EA) that evolves computer programs in any language through CFGs represented in BNF grammars.
Since EAs are computationally demanding, sensible utilisation of computational resources, especially when problems scale up, is essential.

Aim
This paper proposes to identify and call the evolutionary runs that are unlikely to produce solutions of acceptable quality.
Use Ant Colony Optimization (ACO) [1] to predict the failure of a GE run.
Analyze and hand-tune the ACO produced predictive models.
Then, use the predictive model to terminate potentially poor runs very early (10 generations).

Conclusion
We improved the solution quality of GE runs using a completely novel prediction approach. This also has significantly reduced the time spent on executing GE. We will focus on further improving the prediction through rapid retraining.

References

Acknowledgements
This work has been supported by the Science Foundation of Ireland (SFI). Grant No. 1 O/IN.1/1031.

The Run Prediction Model (RPM)
The Run Prediction Model (RPM) + GE armed with a rule based model that predicts the quality of each run.
Figure 1 presents the block diagram of RPM+GE.
The predictive model notes the changes in 4 parameters over first 10 generations: best fitness (BFC), average fitness (AFC), average actual length (AALC), average effective length (AELC).
We subjectively select a different threshold for each problem as a measure of acceptable quality.
The predictive model then judges whether a GE run can cross that threshold; hence, a binary classification of each run.

Results
Table 1 presents the experimental problems and the data sets used.
Data sets are classified based on a predefined success threshold.
Defined a separate success threshold for each and every problem as the solution producing ability of GE is unique on each problem.
Greatly improved the solution quality rather the computational effort is high for data set preparation.

Table 1: Experimental problems and the classification of the data sets.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Class</th>
<th>Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>f₁ = (1 + x)²</td>
<td>590</td>
</tr>
<tr>
<td></td>
<td>f₂ = x²−x²+y²−y</td>
<td>606</td>
</tr>
<tr>
<td></td>
<td>f₃ = x²−y²−x</td>
<td>848</td>
</tr>
<tr>
<td></td>
<td>f₄ = x³</td>
<td>518</td>
</tr>
</tbody>
</table>

Figure 2: Comparison of end of run fitness results both in standard and RPM+GE.

<table>
<thead>
<tr>
<th>Problem</th>
<th>GE</th>
<th>RPM+GE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Fruitful Runs</td>
<td>Fruitful Runs</td>
</tr>
<tr>
<td></td>
<td>f₁</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>f₂</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>f₃</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>f₄</td>
<td>5</td>
</tr>
</tbody>
</table>

Table 2: Success rate of both the approaches.

Figure 1: Block diagram of the Run Prediction Model (RPM) applied GE.