Appendix: Efficient Implementation of LIBRARYSEARCH

Tomasz P. Pawlak, Bartosz Wieloch, Krzysztof Krawiec, *Member, IEEE*

We present here the implementation details of the LIBRARYSEARCH algorithm, presented in Section V-D of [1], that finds in the library of programs the program that has the smallest semantic distance to the desired semantics. There are two phases of the library search process. First, we look for a program in the library L that minimizes Formula 4. Then, we calculate a constant according to Formula 5. If (5) is smaller than (4), LIBRARYSEARCH returns the calculated constant, otherwise it returns the program found in the library.

The brief notation in Formula 4 may suggests that the algorithm has to consider all combinations y of the desired outputs from D_is, and thus may be subject to combinatorial explosion. However, for the domains considered in this paper, d is a Minkowsky norm and thus it is monotonic with respect to differences on each dimension. Therefore:

$$\arg\min_{p\in L} \min_{y\in D_1\times\ldots\times D_n} d\left(y, s(p)\right) \tag{6}$$

$$= \arg\min_{p\in L} \min_{y\in D_1\times\ldots\times D_n} \left[\sum_{j=1}^{n} |y_j - s(p)_j|^w\right]^{\frac{1}{w}} \tag{7}$$

$$= \arg\min_{p\in L} \left[\min_{y\in D_1\times\ldots\times D_n} \sum_{j=1}^{n} |y_j - s(p)_j|^w\right]^{\frac{1}{w}} \tag{8}$$

$$= \arg\min_{p\in L} \min_{y\in D_1\times\ldots\times D_n} \sum_{j=1}^{n} \min_{y_k\in D_j} |y_k - s(p)_j|^w \tag{9}$$

$$= \arg\min_{p\in L} \sum_{D_i\in D} \left(\min_{y_j\in D_i} |y_j - s(p)_j|\right)^w \tag{10}$$

Starting from Formula 4, we substitute d with the Minkowsky distance (7). Since the $\frac{1}{w}$ exponentiation is monotonous, we can move the exponent outside the $\min$ term (8) and discard it in the next step (9), because this does not change the outcome of $\arg\min$. The minimum of sums is equal to minimum of sums of minimums (9). Hence each sum in (9) is equal, we can drop the outer $\min$ term (10).

Formula 10 proves that the choice of the optimal desired output y_j (i.e., such that diverges the least from the jth component of program semantics, $s(p)_j$) can be conducted for each D_i independently. This can be done in a linear time with respect to the number of desired values in each D_i.

The authors are with the Institute of Computing Science, Poznan University of Technology, Poznań, Poland, e-mails: tpawlak@cs.put.poznan.pl, bwieloch@cs.put.poznan.pl, krawiec@cs.put.poznan.pl.

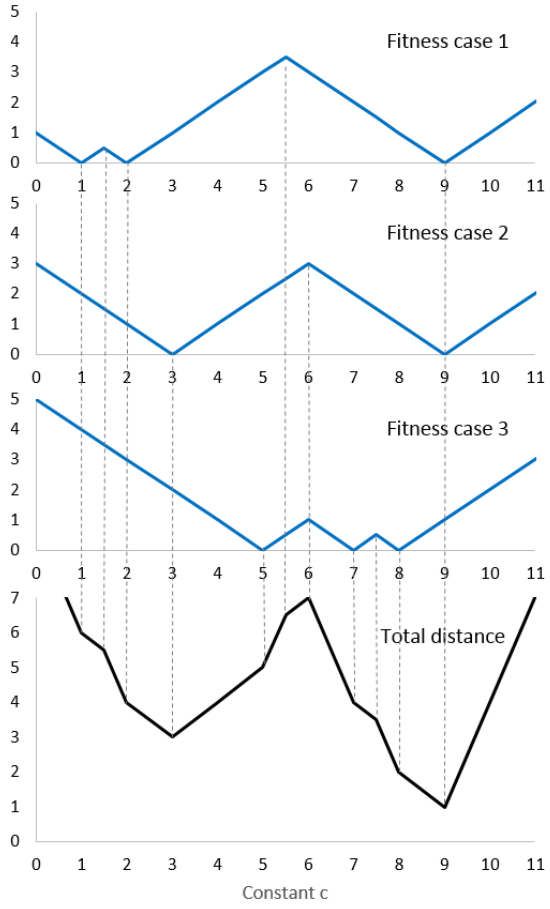

Figure 6. Exemplary three-dimensional desired semantics $D = (\{1,2,9\},\{3,9\},\{5,7,8\})$. The upper three plots present the marginal distance functions for each D_i. The last plot shows the overall distance (Formula 11). The minimum in $c = 9$ is the global minimum of (11), i.e., the optimal constant.

Sorting D_is (carried out once, before LIBRARYSEARCH call) allows using binary search to choose y_j, so this first phase of LIBRARYSEARCH operates in $O(|L|\sum_{D_i\in D}\log(|D_i|))$ time.

Based on an analogous reasoning, we can transform Formula 5 that defines the minimization problem for constants, which leads to (for symbolic regression):

$$\arg\min_{c\in\mathbb{R}} \sum_{D_i\in D} \min_{y_i\in D_i} |y_i - c| \tag{11}$$

$$= \arg\min_{c\in D_k : D_k\in D} \sum_{D_i\in D} \min_{y_i\in D_i} |y_i - c| \tag{12}$$

This transformation is based on the observation that each set of desired values D_i defines its own 'marginal' distance

function (the term under the summation) that is piecewise linear, nowhere constant and has minima in points from D_i (where the $\min$ term attains zero; Fig. 6). Since a sum of such functions is also a piecewise linear function, the overall distance ((11), the bottom plot in Fig. 6) has local minima only for such cs where at least one of the marginal functions reaches zero. Sketch of proof: Between the breakpoints, the total distance is a linear function of c so it cannot have a local minimum. The slope α of that function is the sum of slopes α_i of the marginal functions. A local minimum is the point at which α changes from negative to positive, which requires at least one α_i to change from negative to positive, i.e., the ith marginal function to have a minimum.

Formula 12 can be calculated in $O(\sum_{D_k} |D_k| \times \sum_{D_i} \log(|D_i|))$ time, assuming that D_is are sorted.

Acknowledgments. Work supported by Polish National Science Centre, T. Pawlak is supported by grant no. DEC-2012/07/N/ST6/03066, B. Wieloch and K. Krawiec are supported by grant no. DEC-2011/01/B/ST6/07318.

REFERENCES

[1] T. P. Pawlak, B. Wieloch, and K. Krawiec, "Semantic backpropagation for designing search operators in genetic programming" *IEEE Transactions on Evolutionary Computation*, 2014.