Optimal Recombination Problem in Genetic Programming for Boolean Functions

Aleksey Zakharov

Sobolev Institute of Mathematics SB RAS, Omsk, Russia

NUMTA–2023

The research was supported by Russian Science Foundation grant N 22-71-10015

Optimization problems with tree based solutions

- problems of constructing nonlinear models (mathematical expressions, functions, algorithms, programs) based on given experimental data, set of variables, basic functions and operations
- decision trees construction
- pattern recognition in protein families and other biosequences

Genetic programming

In a genetic programming algorithm, a population of trees is iteratively transformed by means of reproduction operators similar to the selection, crossover (recombination), mutation and local improvements in wildlife and societies[a].

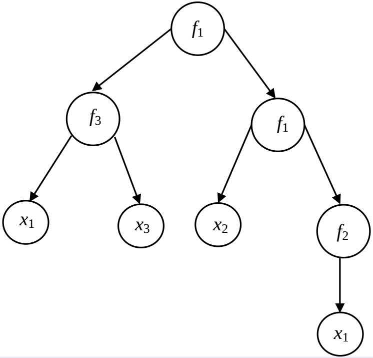
[a] Koza J.R., Poli R.: Genetic programming (2005)

References

1. Koza J. R. (1992) Genetic Programming: On the Programming of Computers by Natural Selection. MIT Press, Cambridge, MA, USA.
2. Koza J.R., Poli R. Genetic Programming. In: Burke E.K., Kendall G. (eds) Search Methodologies. Springer, Boston, MA (2005).
3. Poli, R., Page J. Solving High-Order Boolean Parity Problems with Smooth Uniform Crossover, Sub-Machine Code GP and Demes. Genetic Programming and Evolvable Machines 1, 37-56 (2000).
4. Langdon W.B. Size Fair and Homologous Tree Crossovers for Tree Genetic Programming. Genetic Programming and Evolvable Machines 1, 95-119 (2000).
5. Moraglio A., Krawiec K., Johnson C.G. Geometric Semantic Genetic Programming. PPSN 2012. Lecture Notes in Computer Science, vol 7491. Springer, Berlin, Heidelberg (2012).
6. Santoso L.W., Singh B., Rajest S.S., Regin R., Kadhim K.H. A Genetic Programming Approach to Binary Classification Problem. EAI Endorsed Transactions on Energy Web. V. 8 (31) (2020).
7. Doerr B., Lissovoi A., Oliveto P. S. (1+ 1) genetic programming with functionally complete instruction sets can evolve Boolean conjunctions and disjunctions with arbitrarily small error. Artificial Intelligence, 319, 103906 (2023).

Solution representation

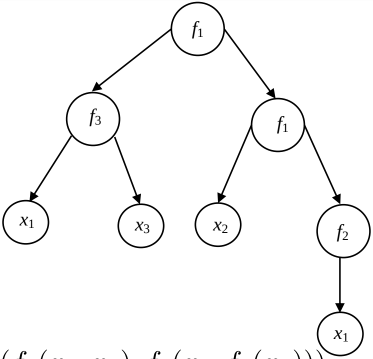
Functional tree $T = (V, E)$.
Leaves contain variables from set $X = \{x_1, x_2, \dots, x_m\}$.
Nodes contain basic functions $\mathcal{F} = \{f_1, f_2, \dots, f_k\}$.



$X = \{x_1, x_2, x_3\}$,
$\mathcal{F} = \{f_1, f_2, f_3\}$

Optimization problem

Input: set of pairs $\{(\bar{x}^i, y^i)\}$, $\bar{x}^i = (\bar{x}^i_1, \ldots, \bar{x}^i_m)$, $i = 1, \ldots, n$.
n is the size of training set.
The objective function $g(T) = \sum_{i=1}^{n} (y_i - T(\bar{x}^i_m))^2$,
$T(\bar{x}^i)$ is the value of functional on the tree T by $\bar{x}^i$



$T(x_1, x_2, x_3) = f_1(f_3(x_1, x_3), f_1(x_2, f_2(x_1)))$

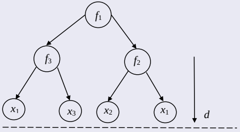
Evolutionary algorithm (genetic programming)

1: Construct an initial population of individuals.
2: Repeat Steps 3–7 until the stopping criterion is satisfied.
3: Select two individuals T_1, T_2 from the current population.
4: Apply mutation operator to both individuals T_1, T_2 with some probability and get individuals T_1', T_2' respectively.
5: Construct an offspring T' by applying crossover operator to the individuals T_1', T_2'.
6: Choose the best individual T_b among individuals T', T_1 and T_2.
7: Replace the worst individual by T_b.
8: Return the best solution (record) with respect to objective function during the run of algorithm.

Generation of initial population

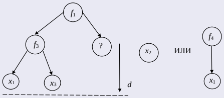
Full method

Full tree of the given depth



Grow method

In each vertex: subtree or leaf with the given possibility
Upper bound on the tree depth

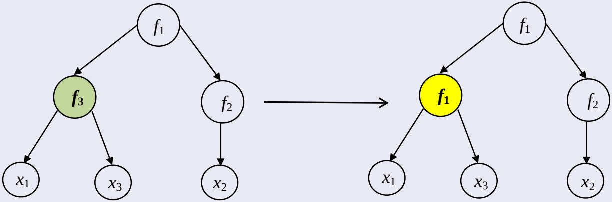


Ramped half-and-half method

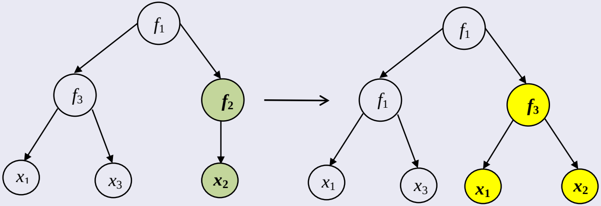
Groups of trees for each depth i: from lower bound to upper bound. Group: 50% of trees by full method with depth i, 50% of trees by grow method with upper bound i. Each group has the same number of elements.

Mutation operators for tree

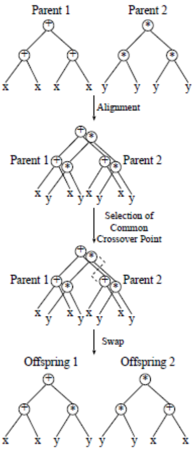
Point mutation (GP-PM)



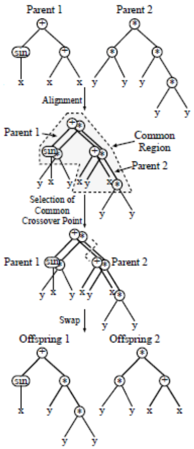
Subtree mutation (GP-SM)



One-point crossover (GP-OPX)



the same shape



general form

Poli R., Langdon W.B. On the search properties of different crossover operators in genetic programming (1998)

Uniform prossover (GP-UX)

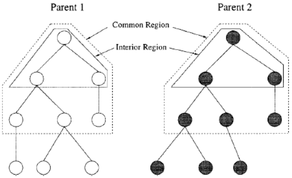


Figure 1. Two parental parse trees prior to GP-UX.

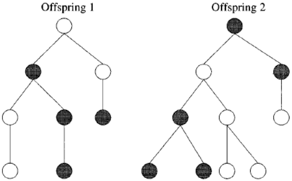


Figure 2. Offspring trees after GP-UX.

Poli R., Page J. Solving high-order Boolean parity problems with smooth uniform crossover, sub-machine code GP and demes (2000)

Optimal recombination problem (ORP)

The definition is based on genes transmission.[1]

Optimal recombination problem

[a] Diven an instance I of combinatorial optimization problem with the set of feasible solutions Sol and two parents $\mathbf{p}^1 = (p_1^1, \dots, p_l^1), \mathbf{p}^2 = (p_1^2, \dots, p_l^2)$ from Sol.
The goal is to find the offspring $\mathbf{p}' \in \mathrm{Sol}$ such that

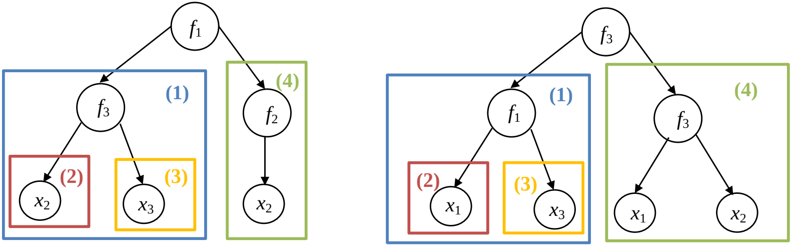
1. $p'_j = p_j^1$ or $p'_j = p_j^2 \quad \forall\, j = 1, \dots, l,$
2. for each $\bar{\mathbf{p}} \in \mathrm{Sol}$ such that $\bar{\mathbf{p}}_j = p_j^1$ or $\bar{\mathbf{p}}_j = p_j^2 \ \forall\, j$ the inequlity holds
$$f(\mathbf{p}') \leq f(\bar{\mathbf{p}})$$
(in case of minimization problem).

[a] A.V. Eremeev, J.V. Kovalenko. Optimal recombination in genetic algorithms for combinatorial optimization problems (2014)

[1] Radcliffe, N.J.: The algebra of genetic algorithms (1994)

ORP on trees

Encoding $\mathbf{p}^1 = (p_1^1, \ldots, p_l^1), \mathbf{p}^2 = (p_1^2, \ldots, p_l^2)$ is referenced to pairs of common nodes, that could be swapped.



Considered crossovers

Optimized one-point: 4 feasible offspring (1), (2), (3), (4).
Optimized uniform: 2^3 feasible offspring, all possible combinations of (2), (3), (4).

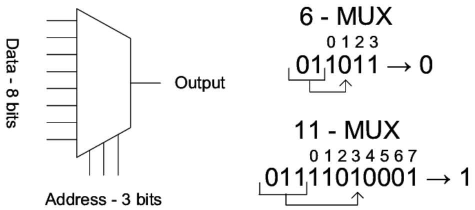
Experiment on Boolean trees

Tree $T = (V, E)$

Leaves contain variables from the set $X = \{x_1, x_2, \ldots, x_m\}$, $x_i \in \{0, 1\}$, $i = 1, 2, \ldots, m$.
Basic functions $\mathcal{F} = \{\wedge, \vee, \neg\wedge, \neg\vee\}$.

Test instances

Truth table of functions
1. even-4-parity (even-4). The value of even-parity function equals 1, iff the input tuple has even number of 1.
2. 6-multiplexor (6-mux).



Experiment on Boolean trees

Initial population: grow or ramped half-and-half (RHH).
Tournament selection.
Mutation: point or subtree mutation.
LS: first improvement.
Crossovers:
randomized: R–OPX, R–UX,
optimized: O–OPX, O–UX.

Parameters of the algorithm

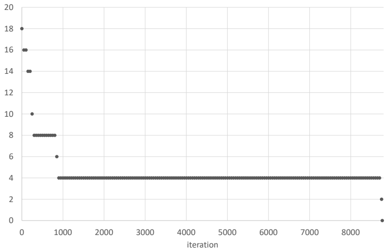
Population size = 100, 30 runs.
Init: low bound 2 and upper bound 8.
Restart of algorithm.
Objective function evaluations: 15000000 for 6-mux, 30000000 for even-4.
The algorithm stops if the optimum is found or the upper bound on the objective function evaluations is reached.

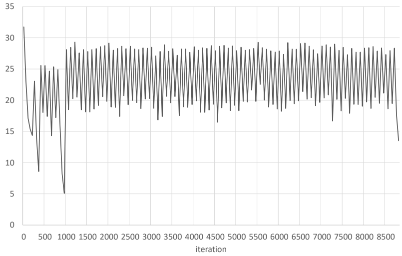
Results of experiment

Problem	Init	Mutation	Crossover	Optimum found (%)	Aver record	Efforts (%)
6-mux	grow	subtree	R–OPX	7	5.27	97
6-mux	grow	subtree	R–UX	0	5.67	100
6-mux	grow	LS + subtree	O–OPX	100	0	9
6-mux	grow	LS + subtree	O–UX	97	0.07	32
6-mux	RHH	point	R–OPX	36.67	1.63	82
6-mux	RHH	point	R–UX	40	1.8	78
6-mux	RHH	LS + point	O–OPX	100	0	3
6-mux	RHH	LS + point	O–UX	96.67	0.03	23
even-4	grow	subtree	R–OPX	0	2.5	100
even-4	grow	subtree	R–UX	0	2.43	100
even-4	grow	LS + subtree	O–OPX	0	1.57	100
even-4	grow	LS + subtree	O–UX	0	2.27	100
even-4	RHH	point	R–OPX	17	0.83	90
even-4	RHH	point	R–UX	37	0.63	83
even-4	RHH	LS + point	O–OPX	100	0	18
even-4	RHH	LS + point	O–UX	87	0.13	51

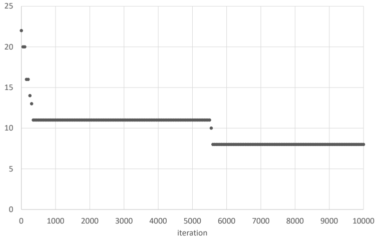
Results of experiment, 6-mux

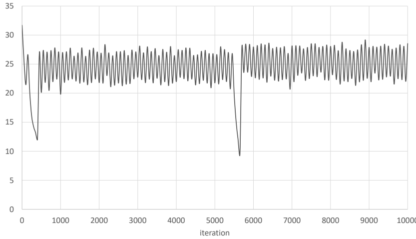
LS (subtree mutation) and O–UX





subtree mutation and R–UX





Conclusions and further research

Conclusion

1. We consider the approximation problem, where solution is represented by tree.
2. We investigate the Optimal recombination problem on trees and consider optimized crossover operators corresponded to randomized ones (one-point and uniform).
3. We carried out computational experiment on Boolean test instances: even-4-parity, 6-mux. Optimized operators show better results compare to randomized ones.

Further research

1. Considering high-dimensional even-parity and multiplexor problems in the context of optimized crossover operators.
2. Constructing the procedure of reducing the objective evaluations.
3. Applying local search procedure to the initial population could give the corresponding performance to the algorithm.

Thank for your attention!

https://gitlab.com/alex2108/tree-crossover