10th International Research/Expert Conference
"Trends in the Development of Machinery and Associated Technology"
TMT 2006, Barcelona-Lloret de Mar, Spain, 11-15 September, 2006

SETTING AN OPTIMAL TRAJECTORY BY MEANS OF ANALYTIC PROGRAMMING

Zuzana Oplatková, Ivan Zelinka
Faculty of Applied Informatics, Tomas Bata University in Zlín
Nad Stráněmi 4511, Zlín
Czech Republic
{oplatkova,zelinka}@fai.utb.cz

ABSTRACT
This paper deals with a novelty tool for symbolic regression, Analytic Programming (AP), which is able to solve various problems from the symbolic regression domain. One of tasks for it can be setting an optimal trajectory for artificial ant on Santa Fe trail which is main application of Analytic Programming in this paper. In this contribution main principles of AP are described and explained. In the second part of the article how AP was used for setting an optimal trajectory for artificial ant according the user requirements is described in detail. An ability to create so called programmes, as well as Genetic Programming (GP) or Grammatical Evolution (GE), is shown in that part. AP is a superstructure of evolutionary algorithms which is necessary to run AP. In this contribution two evolutionary algorithms – Simulated Annealing and Differential Evolution were used to carry preliminary simulations out.
Keywords: Analytic Programming, symbolic regression, evolutionary algorithms

1. INTRODUCTION
This contribution demonstrates use of method which is independent on computer platform – Analytic Programming. This tool is able to synthesize a new program from the basic set of elementary functions as symbolic regression. The aim is to find an analytical formula which corresponds with required tasks. The principles of this method can be found in [1 - 4].

2. PROBLEM DESIGN

2.1. Santa Fe trail
The Santa Fe trail, demonstrated in Fig. 3, was chosen from [5] to make a comparative study with the same problem which was solved by Koza in Genetic Programming [6].

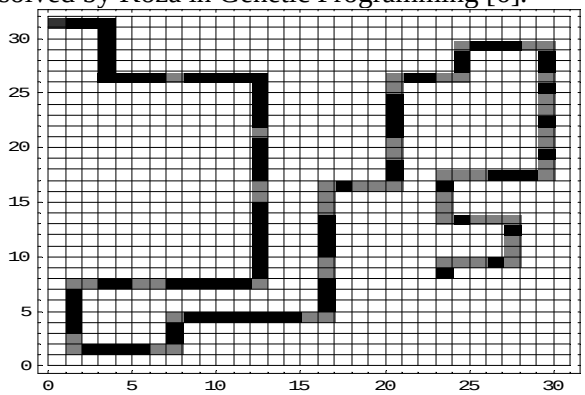


Figure 1: Santa Fe trail

The aim of the task is that an artificial ant should go through defined trail and eat all food what is there.
The SantaFe trail is defined as a 32 x 31 fields where food is set out. In Fig. 1 a black field is food for the ant. The grey one is basically the same as a white field but for clearness was used the grey colour. The grey fields represent obstacles (fields without food on the road) for the ant. If there would not be these holes the ant could go directly through the way. It would be enough to go and see before ant if there is food. If yes ant would go straight and eat the bait. If not it would turn around and see where is food and the cycle would repeat till the ant would eat the last bait.
But in real world robots have obstacles in their moving. Therefore also in this case such approach was chosen. The first problem which ant has to overcome is the simple hole (position [8,27] in Fig. 3). Second one is two holes in the line (positions [13,16] and [13,17]), or three holes ([17,15], [17,16], [17,17]). Next problem is holes in the corners – one (position [13,8]], two ([1,8],[2,8]) and three holes ([17,15], [17,16], [17,17]).

2.2. Set of functions

The set of functions used for movements of the ant is following. As a set of variables GFS0 [4], i. e. in the case of this article functions, which provide moving of an ant, without any argument which could be add during the process of evolution.
The set consist of
GFS0 = {Left, Right, Move},
Where
GFS0 – a set of variables and terminals [4]
Left – function for turning around in the anticlockwise direction
Right – function for turning around in the clockwise direction
Move – function for moving straight and if a bait is in the field where the ant is moved, it is eaten.

This set of functions is not enough to make successfully a desired task. More functions are necessary. Then a GFS2 and GFS3 were set up.
GFS2 = {IfFoodAhead, Prog2}
GFS3 = {Prog3}
Where the number in GFS means the arity of the functions inside, i.e. number of arguments which are needed to be evaluated correctly. Arguments are added to those functions during evolution process [8].
IfFoodAhead is a decision function – the ant controls the field in front of it and if there is food, the function in the field for truth argument is executed, otherwise function in false position.
Prog2 and Prog3 are the same function in the principle. They do 2 or 3 functions in the same time.
These two functions were originally defined also in Koza's approach [6] but in AP it is necessary because of structure of generating the program.

2.3. Fitness function

The aim of the ant is to eat all food on the way. There are 89 baits. This is so called raw fitness. And the value of cost function (1) is calculated as a difference between raw fitness and a number of baits eaten by an ant [1] which went through the grid according to just generated way.

$$CV = 89 - NumberFood \quad (1)$$

NumberFood - number of eaten baits by an ant according to synthesized way

The aim is to find such formula whose cost value is equal zero. To obtain an appropriate solution two constraints should be set up into a cost function. One is a limitation concerned to number of steps. It is not desired ant to go field by field in the grid. A requirement to the fastest way and the most effective is desired. Then a limit of steps was equal to 600.

2.4. Used evolutionary algorithms

In this paper Differential Evolution (DE) and Simulated Annealing (SA) were used as evolutionary algorithms. For detailed information see – [7, 8]

3. EXPERIMENTAL RESULTS

The main idea is to show that DE and SA are able to solve such problems of symbolic regression – setting a trajectory – by means of Analytic Programming.

As a reason of time consuming simulations until the printing this article were done 15 simulations for each algorithm.

DE has all simulations with positive result. SA was not so successful, only 8 positive results. Simulations are still in process.

The equation 2 shows one example of generated programs which was successful. All food was eaten. It was the fastest way of the ant but one of longest records from number of commands point of view. As simulations showed it can be stated that the smallest number of commands does not have to cause the smallest number of steps. And vice versa, that small number of steps does not mean the small set of commands.

The results are concerned to cost function evaluation firstly. As you can see in Table 1 the lowest number of cost function evaluations equals to 2 697 for SA and 9493 for DE.

```
IfFoodAhead IfFoodAhead Move,
  IfFoodAhead Move, IfFoodAhead Left, Move    ,
 Prog3 Right, IfFoodAhead Move,
   Prog3 IfFoodAhead Move, Right , Right,
    IfFoodAhead Move, IfFoodAhead Prog2
       IfFoodAhead Right, IfFoodAhead IfFoodAhead
          Prog2 Move, Prog3 IfFoodAhead
             Prog3 Right, Left, Left , Right ,
            IfFoodAhead Move, Right , Prog2 Left,
             Left    , Move , Prog2 Move, Move    ,
      Prog2 Right, Move   , Right     , Move
```

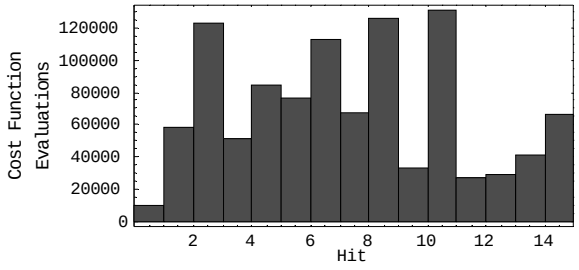
(2)

Table 1: Cost function evaluation for DE and SA

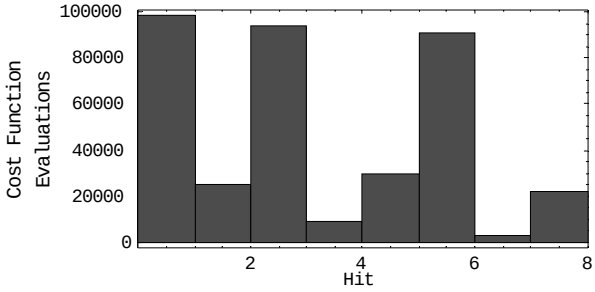
	Cost Function Evaluation	
	DE	SA
Minimum	9 493	2 697
Maximum	131 156	98 241
Average	69176	46 483

Second indicator depicts histogram of successful hits and the number of cost function evaluations for each hit – Fig. 2 - 3. There are not included negative results.

Next point which we were interested in was a number of commands for the ant and number of steps required to eat all baits (Table 2 - 3).



Figures: 2 - Histogram of DE algorithm



Figures 3: Histogram of SA algorithm

Table 2 – Number of commands

	Number of leaves (commands)		
	SOMA	DE	SA
Minimum	11	14	15
Maximum	50	50	50
Average	33	31	29

Table 3 – Number of steps

	Number of steps		
	SOMA	DE	SA
Minimum	396	387	406
Maximum	606	597	605
Average	559	518	556

The minimal value of steps was 387. This result was found by DE algorithm.

4. CONCLUSIONS

This contribution deals with a tool for symbolic regression. This study shows that this tool is suitable not only for mathematical regression but also for setting of optimal trajectory for artificial ant which can be replaced by robots in real world, in industry.

To compare with standard GP, which is also a tool for symbolic regression [6], it can be stated on the basic results above that AP can solve this kind of problems in shorter times as cost function evaluations are counted. The aim of this study was not to show that AP is better or worse than GP (or GE when compared) but that AP is also a powerful tool for symbolic regression with support of different evolutionary algorithms.

The main object of the paper was to show that symbolic regression done by AP is able to solve also cases where linguistic terms as for example commands for movement of artificial ant or robots in real world are. Here simulations for 2 algorithms – DE and SA were done.

Reached results –15 from 15 for DE and 8 from 15 for SA which accomplished the required tasks thus Analytic Programming is able to solve such kind of problems in symbolic regression. It is supposed that the cost function is very complicated with quite a lot of local optima and therefore the Simulated Annealing was not successful as DE was.

Future research is key activity in this field. The following steps are to finished simulations with GA and to try some other class of problems to show that Analytic Programming is powerful tool as Genetic Programming or Grammatical Evolution are.

5. ACKNOWLEDGEMENTS

This work was supported by the grant NO. MSM 7088352101 of the Ministry of Education of the Czech Republic and by grants of Grant Agency of Czech Republic GACR 102/06/1132 and GACR 102/05/0271.

6. REFERENCES

[1] Zelinka I. 2002: Analytic programming by Means of Soma Algorithm. Mendel ’02, *In: Proc. 8th International Conference on Soft Computing Mendel’02*, Brno, Czech Republic, 2002, 93-101., ISBN 80-214-2135-5

[2] Zelinka I., Oplatkova Z. 2003: Analytic programming – Comparative Study. *CIRAS’03, The second International Conference on Computational Intelligence, Robotics, and Autonomous Systems*, Singapore, 2003, ISSN 0219-6131

[3] Zelinka I.,Oplatkova Z, Nolle L. 2004b: Boolean Symmetry Function Synthesis by Means of Arbitrary Evolutionary Algorithms-Comparative Study, ESM '2004, I*n: Proc. 18th European Simulation Multiconference*, Magdeburg, Germany 2004

[4] Zelinka I.,Oplatkova Z, Nolle L. 2005: Boolean Symmetry Function Synthesis by Means of Arbitrary Evolutionary Algorithms-Comparative Study, *International Journal of Simulation Systems, Science and Technology*, *Volume 6*, Number 9, August 2005, pages 44 - 56, ISSN: 1473-8031, online http://ducati.doc.ntu.ac.uk/uksim/journal/Vol-6/No.9/cover.htm, ISSN: 1473-804x

[5] Oplatkova Z. 2005, Optimal Trajectory of Robots Using Symbolic Regression, *In: CD-ROM of Proc. 56th International Astronautical Congress 2005*, Fukuoka, Japan, 2005, paper nr. IAC-05-C1.4.07

[6] Koza J.R.: Genetic Programming, MIT Press, ISBN 0-262-11189-6, 1998

[7] Kirkpatrick S., Gelatt C. D., Vecchi M. P. 1983, Optimization by Simulated Annealing, *Science*, 13 May 1983, *220* (4598), p. 671 – 680

[8] Price K., Storn R. M., Lampinen J. A. 2005: Differential Evolution : *A Practical Approach to Global Optimization* (Natural Computing Series) Springer; 1 edition , 2005, ISBN: 3540209506