Complete Binary Variational Analytic Programming for Synthesis of Control at Dynamic Constraints

Askhat Diveev[1,2], and *Elizaveta* Shmalko[1,2,*]

[1]Federal Research Centre “Computer Science and Control” of Russia Academy of Sciences, 119333Moscow, Russia
[2]RUDN University, Engineering Academy, 117198 Moscow, Russia

Abstract. Dynamic constraints arise at control of the robot moving in the neighborhood of other robots. In this case, together with ensuring stability of the robot in a goal point it is necessary to provide instability for the robot in the area of state space where is some other robot. For the solution of this task, we use a new method of symbolic regression - a method of binary variational analytical programming. The new method has advantages in comparison with genetic or analytical programming in case of performing crossover, and it uses the principle of small variations of the basic solution to increase the efficiency of an algorithm of search of the optimal solution.

1 Introduction

Solution of control tasks for robot movement originally involves providing stability of the robot concerning the set of points in state space, then building trajectories of robots movements. The trajectories consist of points of state space. To design trajectories, engineer must consider various constraints including the dynamic constraints arising because of presence of other robots in the neighbourhood of the trajectory. The problem of control of movement of stable robot concerning some set of points in a state space does not cause big difficulties. It is enough to set a point in the state space, and the robot because of property of stability itself will be going to it. Difficulties in control arise when the constraints in the neighbourhood of the robot appear. The most complex problem arises when some other moving robot creates the constraints. If we know the paths of the other robot, then to avoid collisions we can construct such trajectories of stable points in the state space that both robots can move without collisions among them. The most complex problem arises when in the neighbourhood of the robot there appears another robot with unknown direction of the movement. In this case, creation of trajectories of stable points in the state space is impossible because of difficulty to predict real position of other robot. In practice in these cases a control system turns on a special mode which has the main objective to avoid collision with the disturbing object.

In this work for the solution of the stated problem, we use a method of synthesis of “stable-unstable” control. The synthesized control system in the form of feedback function depending on a state of the object has to provide stability of the object concerning one given point of state space and instability concerning other point of the state space. We deliver two types of points of state space to an input of a control system. The points of the first type define the goal of the movement of the robot. The points of the second type determine space coordinates of the disturbing robot. For the second type points, we determine also the area of instability. The synthesized function of control together with the set of two-type points in the state space defines mathematical model of the control object in the form of system of ordinary differential equations with special points of two types, a stable point of the goal of the movement and an unstable point to avoid a collision with the disturbing robot. For the second point we set the limited area of instability. To solve the problem of synthesis of stable-unstable control, we use a method of symbolic regression [1-3].

Application of numerical methods of symbolic regression for the solution of the problem of synthesis is already known [3-9]. Unlike the known numerical solutions of the synthesis problem, in the present work we use a new approach of synthesis of stable-unstable control of a plant. Moreover, the known methods of symbolic regression have an essential shortcoming for an algorithm of evolutionary search. In these methods [1-9] at search of the optimal solution by a genetic algorithm, it is necessary to carry out the main crossover operation in certain points, or it is essential to adjust the new solution after accomplishment of the crossover. Use of the principle of small variations [10] of the basic solution, as in variational methods of genetic and analytical programming [11-13], does not significantly change the situation. It is possible to obtain an incorrect code after some variations, and this requires a correction of the code. It leads to consumption of computing time and breaks the principle of small variations.

[*] Corresponding author: e.shmalko@gmail.com

To search for a solution, we use a method of binary variation analytic programming [14] in its complete form. The method codes mathematical expressions in the form of a copmlete binary computing tree and in the computer memory in the form of the ordered limited set of integers. Nodes of the tree-graph are connected with functions with two arguments. Edges of the graph are connected with functions with one argument. Leaves of the tree or source-nodes of the graph are connected with arguments of mathematical expression or with unit elements of functions with two arguments. Functions with one argument includes the identity function. To expand a class of required functions, we add constant parameters to arguments of mathematical expression. The proposed method of complete binary variation analytic programming allows carry out crossover operation in any point and does not demand correction of a code at any variations. We look for values of parameters together with structure of mathematical expression.

2 Problem Statement

Consider formal problem statement of synthesis of stable-unstable control.

Set a model of control object

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x},\mathbf{u}), \tag{1}$$

where $\mathbf{x}$ is a vector of state, $\mathbf{x} \in \mathbf{R}^n$, $\mathbf{x} = [x_1 \ \dots \ x_n]^T$, $\mathbf{u}$ is a vector of control, $\mathbf{u} \in \mathbf{R}^m$, $\mathbf{u} = [u_1 \ \dots \ u_m]^T$, $m \le n$.

Set constraints on the control

$$\mathbf{u} \in \mathrm{U} \subseteq \mathbf{R}^m, \tag{2}$$

where U is a compact set.

Set a terminal point

$$\mathbf{x}^* \in \mathbf{R}^n. \tag{3}$$

Set a limited unstable area

$$\widetilde{\mathrm{X}} \subseteq \mathbf{R}^n \tag{4}$$

and an unstable point

$$\widetilde{\mathbf{x}} \in \widetilde{\mathrm{X}} \subseteq \mathbf{R}^n. \tag{5}$$

It is necessary to find a control in the form of function of space coordinates

$$\mathbf{u} = \mathbf{h}(\mathbf{x}),$$

$$\mathbf{h}(\mathbf{x}) : \mathbf{R}^n \to \mathbf{R}^m, \ \forall \mathbf{x} \in \mathbf{R}^n, \ \mathbf{h}(\mathbf{x}) \in \mathrm{U} \subseteq \mathbf{R}^m. \tag{6}$$

The closed system of the differential equations $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x},\mathbf{h}(\mathbf{x}))$ is locally unstable concerning $\widetilde{\mathbf{x}}$ in unstable area $\widetilde{\mathrm{X}} \subseteq \mathbf{R}^n$ and is stable concerning the terminal point $\mathbf{x}^* \in \mathbf{R}^n$.

For the solution of the task, we use a numerical method of symbolic regression which by means of an evolutionary algorithm will find the function meeting the specified requirements. For numerical assessment of the selected function we use the following criterion:

$$J(\mathbf{x}^0) = \int_0^{t^+} f_0(\mathbf{x}(t), \mathbf{x}^*, \widetilde{\mathbf{x}}) dt, \tag{7}$$

where t^+ is a defined time of control process, $\mathbf{x}^0$ is initial conditions $\mathbf{x}^0 = \mathbf{x}(0)$,

$$f_0(\mathbf{x}(t), \mathbf{x}^*, \widetilde{\mathbf{x}}) = \begin{cases} \| \mathbf{x}(t) - \mathbf{x}^* \|, \text{ if } \| \mathbf{x}(t) - \widetilde{\mathbf{x}} \|_2 > r \\ \| \mathbf{x}(t) - \mathbf{x}^* \| + r - \| \mathbf{x}(t) - \widetilde{\mathbf{x}} \|_2, \text{otherwise} \end{cases},$$

r is a given size of unstable area,

$$\| \mathbf{x} - \widetilde{\mathbf{x}} \|_2 = \sqrt{\sum_{i=1}^{n} (x_i - \widetilde{x}_i)^2}$$

$$\| \mathbf{x} - \mathbf{x}^* \| = \max\{| x_i - x_i^* | : i = 1, \dots, n\}.$$

For ensuring properties of stability and instability, calculate the functional (7) for a given set of initial conditions

$$J = \sum_{i=1}^{M} J(\mathbf{x}^{0,i}) \to \min, \tag{8}$$

where $\mathbf{x}^{0,i}$ is given initial conditions, $i = 1, \dots, M$.

3 Complete Binary Variational Analytic Programming

To build a code of binary analytic programming we use the following basic sets:

- a set of arguments of mathematical expression

$$\mathrm{F}_0 = (q_1, \dots, q_P, x_1, \dots, x_N); \tag{9}$$

- a set of functions with one argument

$$\mathrm{F}_1 = (f_{1,1}(z) = z, f_{1,2}(z), \dots, f_{1,R}(z)); \tag{10}$$

- a set of functions with two arguments

$$\mathrm{F}_2 = (f_{2,1}(z_1, z_2), \dots, f_{2,S}(z_1, z_2)); \tag{11}$$

- a set of unit elements for functions with two arguments

$$\mathrm{E}_2 = (e_1, \dots, e_M). \tag{12}$$

A set of function with one argument must include the identity function

$$f_{1,1}(z)=z\,. \quad (13)$$

Every function with two arguments of the set (11) has unit element from the set (12), $\forall f_{2,i}(z_1,z_2)\in \mathrm{F}_2$ $\exists e_j \in \mathrm{E}_2$

$$f_{2,i}(e_j,z_2)=z_2\,,\; f_{2,i}(z_1,e_j)=z_1\,, \quad (14)$$

where $i\in\{1,\ldots,S\}$, $j\in\{1,\ldots,M\}$.

To generate a code of the binary analytic programming we combine into one ordered set a set of arguments (9) of mathematical expression, a set of unit elements (12)

$$\mathrm{F}=(f_1=q_1,\ldots,f_P=q_P,f_{P+1}=x_1,\ldots,f_{P+N}=x_N,$$
$$f_{P+N+1}=e_1,\ldots f_{P+N+M}=e_M) \quad (15)$$

We write down mathematical expression in the form of composition of nested functions and arguments of mathematical expression

$$y=f_{\alpha_1}(f_{\alpha_2}(\ldots f_{\alpha_K})\ldots)=f_{\alpha_1}\circ f_{\alpha_2}\circ\ldots\circ f_{\alpha_K}\,, \quad (16)$$

where $f_{\alpha_i}\in \mathrm{F}\cup\mathrm{F}_1\cup\mathrm{F}_2$, $i=1,\ldots,K$.

We define the most nesting depth of an element by the composition. Let the most depth is equal L. It means that we have binary computing tree with level L. It has 2^L leaves. We write down mathematical expression in the form of a complete binary computing tree (see Fig.1)

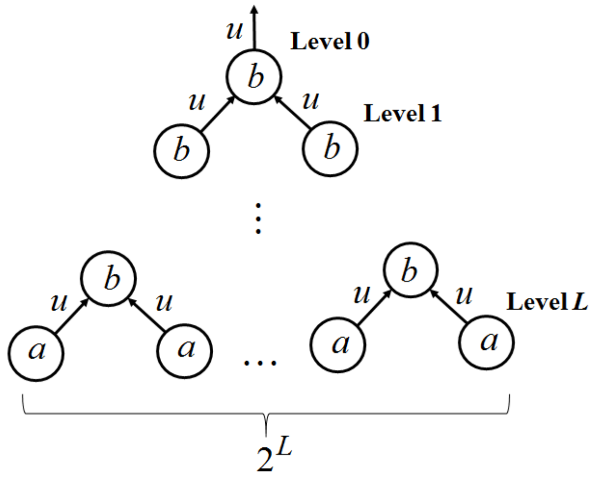

Fig. 1. A comlete binary computing tree

In the Fig.1 a is number of an element of set F (15), b is number of an element of set F_2 (11), u is number of an element of set F_1 (10). Each level includes numbers of elements from the set F_1 and the sane numbers of elements from the set F_2, and the last level includes numbers from the set F. If the tree has redundant nodes or edges, then we use number of any function with two arguments for nodes/ number the identity function for edges and number of unit element for this function with two arguments on the last level. Code of binary variational analytic programming for mathematical expression is an ordered set of numbers of elements from first, second and others levels of the tree

$$\mathrm{C}=(u_1,b_1,u_{2,1},u_{2,2},b_{2,1},b_{2,2},\ldots$$
$$\ldots,u_{L,1},u_{L,2},\ldots,u_{L,2^L},a_{L,1},a_{L,2},\ldots,a_{L,2^L}) \quad (17)$$

4 Variational Genetic Algorithm

To search for optimal solution, we use a variational genetic algorithm. Define a small variation of a code of binary analytic programming as an integer vector of two components

$$\mathbf{w}=[w_1\;\; w_2]^T\,, \quad (18)$$

where w_1 is a position in a code, w_2 is a new value of element of the code.
Let

$$\mathrm{C}=(c_1,\ldots,\,c_K) \quad (19)$$

be a code of mathematical expression for level L. Then

$$K=2^{L+2}-2\,, \quad (20)$$

and we obtain after variation (18) a new code

$$\mathbf{w}\circ\mathrm{C}=(\overbrace{c_1,\ldots,w_2}^{w_1},\ldots,c_K)\,. \quad (21)$$

The small variation (18) satisfies the following conditions

$$w_1\in\{1,\ldots,2^{L+2}-2\}\,, \quad (22)$$

$$w_1\in\begin{cases}\{1,\ldots,|\mathrm{F}_1|\},\text{ if } 2^i-1\le w_1\le 3\cdot 2^{i-1}-2\\ \{1,\ldots,|\mathrm{F}_2|\},\text{ if } 3\cdot 2^{i-1}-1\le w_1\le 2^{i+1}\text{-}2\\ \{1,\ldots,|\mathrm{F}|\},\text{ if } 3\cdot 2^L-1\le w_1\le 2^{L+2}-2\end{cases}, \quad (23)$$

where $i=1,\ldots,L$.

A variational genetic algorithm consists of the following stages:

1) Set a code of basic solution

$$\mathrm{C}_0=(c_1^0,\ldots,\,c_K^0)\,;$$

2) Generate a set of ordered sets of small variations

$$\Omega=\{\mathrm{W}_1,\ldots,\mathrm{W}_H\}\,,$$

where W_1 is an ordered set of variations $\mathrm{W}_i=(\mathbf{w}^{i,1},\ldots,\mathbf{w}^{i,l})$, $\mathbf{w}^{i,j}=[w_1^{i,j}\;\; w_2^{i,j}]^T$, $i=1,\ldots,H$, $j=1,\ldots,l$, l is a set of numbers of variations.

We carry out crossover and mutation on the sets of variations. Select two parents $\mathrm{W}_i = (\mathbf{w}^{i,1}, \ldots, \mathbf{w}^{i,l})$ and $\mathrm{W}_j = (\mathbf{w}^{j,1}, \ldots, \mathbf{w}^{j,l})$, and randomly define a point of crossover $k \in \{1, \ldots, l\}$ and exchange tails of the parents

$$\tilde{\mathrm{W}}_i = (\mathbf{w}^{i,1}, \ldots, \mathbf{w}^{i,k-1}, \mathbf{w}^{j,k}, \ldots, \mathbf{w}^{j,l}),$$

$$\tilde{\mathrm{W}}_j = (\mathbf{w}^{j,1}, \ldots, \mathbf{w}^{j,k-1}, \mathbf{w}^{i,k}, \ldots, \mathbf{w}^{i,l}).$$

Define randomly a mutation point $\mu \in \{1, \ldots, l\}$ and generate new variation in the point μ $\mathbf{w}^{i,\mu} = [w_1^{i,\mu} \ w_2^{i,\mu}]^T$ for one and other new solutions.

5 Computational example

Consider a mobile robot as a control object

$$\dot{x} = u_1 \cos(\theta),$$

$$\dot{y} = u_1 \sin(\theta),$$

$$\dot{\theta} = u_2,$$

where $-1 \le u_{1,2} \le 1$.

It is necessary from some initial conditions

$$\mathrm{X}_0 = \{\mathbf{x}^{0,1} = [-1 \ -2 \ 0]^T, \mathbf{x}^{0,2} = [-1 \ -1.6 \ 0]^T, \ldots$$

$$\ldots, \mathbf{x}^{0,11} = [-1 \ 2 \ 0]^T, \mathbf{x}^{0,12} = [2 \ -2 \ 0]^T, \ldots$$

$$\ldots, \mathbf{x}^{0,22} = [2 \ 2 \ 0]^T\}$$

to find such a control to rich the terminal condition

$$\mathbf{x}^f = [0 \ 0 \ 0]^T$$

with minimal time.

Functionals have the following forms

$$J_1 = \sum_{j=1}^{22} (t_f(\mathbf{x}^{0,j}) + s(\mathbf{x}^{0,j})),$$

where

$$s(\mathbf{x}^{0,j}) = \int_0^{t_j} f_0(\mathbf{x}(t))dt,$$

$$t_j = t(\mathbf{x}^{0,j}),$$

$t(\mathbf{x}^{0,j})$ is a time of control process,

$$t(\mathbf{x}^{0,j}) = \begin{cases} t, \text{ if } \|\mathbf{x}(t) - \mathbf{x}^f\| < \varepsilon \\ t^+ \text{ - otherwise} \end{cases},$$

$$f_0(\mathbf{x}(t)) = \begin{cases} r = \sqrt{(x(t) - \tilde{x}(t))^2 + (y(t) - \tilde{y}(t))^2}, \text{ if } r < r^+ \\ 0 \text{ - otherwise} \end{cases},$$

$(\tilde{x}, \tilde{y})$ are coordinates of unstable point, ε, t^+, r^+ are given positive values, $\varepsilon = 0.01$, $t^+ = 4$. $r^+ = 0.5$.

For calculation, we used the following functions with one argument:

$$\mathrm{F}_1 = (f_1(z) = z, f_2(z) = z^2, f_3(z) = -z,$$

$$f_4(z) = \mathrm{sgn}(z)\sqrt{|z|}, f_5(z) = \frac{1}{z}, f_6(z) = e^z,$$

$$f_7(z) = \ln(|z|), f_8(z) = \frac{1 - e^{-z}}{1 + e^{-z}}, f_9(z) = \begin{cases} 1, \text{ if } z \ge 0 \\ 0 \text{ - otherwise} \end{cases},$$

$$f_{10}(z) = \mathrm{sgn}(z), f_{14}(z) = z^3, f_{15}(z) = \sqrt[3]{z},$$

$$f_{16}(z) = \begin{cases} \mathrm{z}, \text{ if } |z| \le 1 \\ \mathrm{sgn(z)} \text{- otherwise} \end{cases}, f_{17}(z) = \mathrm{sgn}(z)\ln(|z| + 1),$$

$$f_{18}(z) = \mathrm{sgn}(z)\exp(|z| - 1), f_{23}(z) = z - z^3).$$

The set of functions with two arguments were

$$\mathrm{F}_2 = (f_1(z_1, z_2) = z_1 + z_2, f_2(z_1, z_2) = z_1 z_2).$$

A basic solution was

$$u_{1,2} = q_1(x^f - x) + q_2(y^f - y) + q_3(\theta^f - \vartheta) +$$

$$q_4(\tilde{x} - x) + q_5(\tilde{y} - y).$$

where $q_i = 1$, $i = 1, \ldots, 5$.

A genetic algorithm had the following parameters: number of possible solutions in initial set – 1024, number of generations among exchange of the basic solution – 256, number of crossovers in one generation – 256, number of variations for one solution – 8.

We have obtained the following control

$$u_{1,2} = \begin{cases} \mathrm{sgn}(u_{1,2}), \text{if } |u_{1,2}| > 1 \\ \tilde{u}_{1,2} \text{ - otherwise} \end{cases},$$

where

$$\tilde{u}_1 = \mathrm{sgn}(AB)\sqrt{|AB|},$$

$$\tilde{u}_2 = \mathrm{sgn}(\mathrm{sgn}(C + D)\ln(|C + D| + 1) +,$$

$$(G + H)^2)\sqrt{|\mathrm{sgn}(C + D)\ln(|C + D| + 1) + (G + H)^2|},$$

$$A = \mathrm{sgn}(-\Delta_y q_2 q_3 \Delta_\theta)\exp(|-\Delta_y q_2 q_3 \Delta_\theta| - 1),$$

$$B = (\mu(q_4) + q_5 + q_1),$$

$$C=\frac{1-\exp(-(\tilde{\Delta}_x-\tilde{\Delta}_y)\mu(q_5\tilde{\Delta}_y^3))}{1+\exp(-(\tilde{\Delta}_x-\tilde{\Delta}_y)\mu(q_5\tilde{\Delta}_y^3))},$$

$$D=\operatorname{sgn}\left(\operatorname{sgn}(\Delta_x)q_3+q_2\frac{1-\exp(-\Delta_x)}{1+\exp(-\Delta_x)}\right)\times$$

$$\exp\left(\left|\operatorname{sgn}(\Delta_x)q_3+q_2\frac{1-\exp(-\Delta_x)}{1+\exp(-\Delta_x)}\right|-1\right),$$

$$G=\sqrt[6]{\Delta_\theta\ln(|q_3|+1)}\sqrt[3]{-\tilde{\Delta}_x^2},$$

$$H=\sqrt[3]{\operatorname{sgn}(q_5^3-q_5)\sqrt{|q_5^3-q_5|}\ln(|q_2\Delta_y|+1)},$$

$$\mu(z)=\begin{cases}\text{z, if } |z|\le 1\\ \text{sgn(z) - otherwise}\end{cases},$$

$$\Delta_x=x^f-x,\ \ \Delta_y=y^f-y,\ \Delta_\theta=\theta^f-\theta,$$

$$\tilde{\Delta}_x=\tilde{x}-x,\ \tilde{\Delta}_y=\tilde{y}-y,$$

$q_1=2.28516$, $q_2=1.47559$, $q_3=2.87891$, 0,07324 $q_4=0.07324$, $q_5=3.95410$.

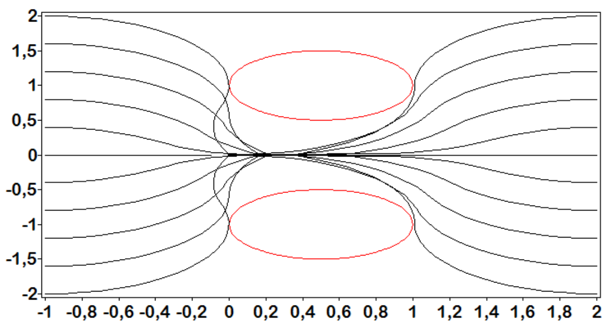
Fig. 2. Movement of robot from 22 intial positions

The fig 2 shows results of simulation of robot movement from twenty two initial positions. There red areas are dynamic phase constraints.

6 Conclusion

The paper presents new method of symbolic regression binary variational analytic programming. A new method codes a mathematical expression in the form of binary computing tree. We have solved the problem of synthesis of control for mobile robot with disturbing phase constraints by the new method. The method has obtained control as a nonlinear complex function depending on the space state coordinates of object and disturbing dynamical phase constraints.

This research was supported by Russian Foundation for Basic Research according to the research project №. 14-08-00008-a and №. 16-29-04224-ofi_m.

References

1. M. O'Neill and C. Ryan, IEEE Trans. Evol. Comp. **5**, 349 – 358 (2001)
2. I. Zelinka, Proc. of 1st Int. Conf. on New Trends in Physics'01, 210–214 (2001)
3. A. Diveev, J. of Comp. and Syst. Sci. Int. **51**(2), 228–243 (2012)
4. K. Cpałka, K. Łapa, A. Przybył, G. Eason, Inf. Tech. And Contr., **44**(4), 433-442 (2015)
5. J. Imae, Y. Kikuchi, N. Ohtsuki, T. Kobayashi, G. Zhai, Proc. of 43rd IEEE Conf. on Decision and Contr., 2734-2739 (2004)
6. R.A. Maher, M.J. Mohamed, Intel. Contr. and Autom., **4**, 94-101 (2013)
7. A. Diveev, E. Sofronova, Proc. 17th IFAC World Congr., 6106 – 6113 (2008)
8. J. Yu, M.A. Keane, J.R. Koza, Proc. of 2000 IEEE Int. Symp. on Comp.-Aided Contr. Syst. Design, 234 – 242 (2000)
9. A.I. Diveev, E.Yu. Shmalko, Proc. of 3rd Int. Conf. on Contr., Decision and Inf. Tech. (CoDIT'16), 077-082 (2016)
10. A.I. Diveev, Proc. of 16th IFAC Workshop on Contr. App. of Optimization, 28-33 (2015)
11. S.I. Ibadulla, E.Yu. Shmalko, K.K. Daurenbekov, Procedia Comp. Sci., **103**, 155-161 (2017)
12. A. Diveev, S. Ibadulla, N. Konyrbaev, E. Shmalko, IFAC-PapersOnLine, **48**(19), 106-111 (2015)
13. A. Diveev, S. Ibadulla, N. Konyrbaev, E. Shmalko, IFAC-PapersOnLine, **48**(19), 113-118 (2015)
14. A.I. Diveev, N.B. Konyrbaev, E.A. Sofronova, Procedia Comp. Sci., **103**, 597-604 (2017)