DEVELOPMENT OF THE REDUCED FINITE ELEMENT MODEL OF BALLISTIC IMPACT AGAINST WOVEN TEXTILE TARGET

Vidmantas Rimavicius, Ramute Barauskiene, Rimantas Barauskas

Kaunas University of Technology, Department of System Analysis, Studentu str. 50-407, Kaunas, Lithuania, vidmantas@achema.com, ramute.barauskiene@ktu.lt, rimantas.barauskas@ktu.lt

Abstract. The work demonstrates a formal approach of creating a reduced cost-effective computational model of the complex physical phenomena of high velocity impact of a missile upon the woven textile target. A model earlier developed by us in LS-DYNA finite element system has been employed as a reference model. The internal structure of the reference model was based on the mathematical representation of all contact interactions among the yarns of the woven structure. The reference model required time-consuming calculations, which were carried out by means of the explicit time integration scheme in LS-DYNA. The internal structure of the model synthesized in this work by using ANSYS finite element system was essentially simpler as it presented the woven structure by means of the girder, the interactions between nodal points of which were expressed by using elastic, viscous and friction link finite elements. Only contact interactions of the surface of the missile against the girder needed to be determined during each time integration step. The mathematical expression of mutual adequacy of the two models was based on the comparison of the response of the reduced model against the response of the reference model by means of appropriately constructed penalty functions. The task of the synthesis of the reduced model has been presented as an optimization problem, the optimization parameters of which were unknown parameters of the reduced model such as equivalent geometric and physical parameters of the rods of the girder and link element characteristics.

Keywords: finite element models, reduction, parameter identification.

1 Introduction

Finite element techniques in principle enable to analyse structures of any level of complexity, including their essentially non-linear behaviour, peculiarities of internal texture, etc. [4]. However, highly adequate models are often generated on expense of very complex structures, huge dimensionalities and internal interactions, which require very large and often prohibitive amounts of computational resources. Building simplified (reduced) computational models is a common practice enabling to obtain solutions with practically acceptable costs.

Ballistic protection models of textiles are based on the analysis of high-velocity impact, which is concentrated in a comparatively small zone of a fabric. However, for the correct representation of the process, large pieces of cloths and packages have to be modelled [1, 2]. For this purpose, the woven structure can be represented by using models of different levels of detalization. Woven structure composed of shell elements [2], simpler and more efficient combined particles model [4], orthotropic membrane models [1], have been employed in order to represent the dynamic behaviour of textile cloths under conditions of mechanical impact and penetration. A special attention and prospective deserve models in which central and distant zones of the same structure are presented by different models. As in [1], the zone of ballistic interaction of textile structure has been modeled by using the complex contact model of a woven structure, meanwhile the distant zones have been presented by membrane elements. The coupling between the zones has been implemented by means of tie constraint [1, 5]. The main purpose of this combination was to implement the “almost infinite” surrounding.

As a rule, such combined models are obtained by using a lot of engineering intuition and basing on profound knowledge of physical properties of the investigated phenomena. More regular approaches are necessary, which enable to synthesize simplified or reduced models of internally complex structures. The parameters of the reduced model can be adjusted by performing the minimization of error functions, quantitatively indicating the non-coincidence of the response between the simplified and reference models.

In this work, we demonstrate a formal approach of creating a reduced cost-effective computational model of the complex physical phenomena of high velocity impact of a bullet upon the woven textile target. Novelty elements exist in offered combination of two different resolution models in a single structure, which was used to create original finite element model of textile fabric ballistic interaction.

2 Problem formulation

Finite element model is used in simulation of various physical phenomena. Simulation with sufficient accuracy requires using models with large number of finite elements which results in enormous scope time-consuming calculations. Even powerful software tools such as ANSYS, LS-DYNA, etc., do not allow to simulate phenomena with required accuracy in acceptable time. The program ANSYS is intended for system

analysis (static and dynamic calculations of solid, liquid bodies and of other mechanical tasks) using finite element method. This software is powerful but simple to use. Simulation of physical systems and high-speed processes (eg., simulation of ballistic processes), calculations and result visualization are implemented using simulation program LS-DYNA.

Effective models of numerical or real experiment are understood as accurate representations of behaviour of reference model and not requiring large calculation resources. They also enable execution of numerical research by using conventional calculation means available to researcher.

We are analysing complex finite elements' computational structure for ballistic contact interaction of a bullet against a textile structure [3], which was implemented in LS-DYNA environment. Accurate numerical solution obtained in LS-DYNA environment is assumed as known, because we can perform individual numerical experiment, i.e. numerical experiment of the lead bullet at the speed $\sim 300\frac{m}{s}$ and a textile perforation. Reference computational model when a bullet penetrates a textile (see Fig.1) was created on the basis of real experiments, and physical parameters were selected empirically.

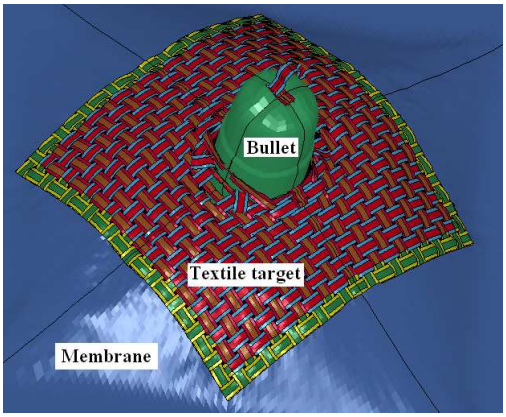


Figure 1. Reference model when a bullet penetrates textile target

Each textile is made of certain linear densely woven yarns. Yarn is composed of filaments, whose number ranges from several tenths to several thousands. Implementation of a textile structure at the level of filaments is complicated due to limited resources, therefore primary component in the textile structure is supposed to be a yarn.

Textile weave is obtained by solving *contact interaction balance* task in reference computational model. Yarns of the direction Ox are elastically deformed by moving their nodes in the direction Oz perpendicular to structure's plane in order to compose yarns of the direction Oy. Model's yarns are in balance after activation of *contact search algorithm*. After balance is maintained, elastic tensions and loads are fully or partially removed in order to simulate multiyarn fibre textile. Therefore similar structure to real textile is obtained. Yarn bending rigidity was supposed to be insignificant in the model. Textile areas distant from the impact area were simulated using elements of orthotropic membrane.

Modelling of computational structure when a bullet penetrates textile is difficult therefore we will model the case when a bullet does not penetrate a textile (see Fig.2).

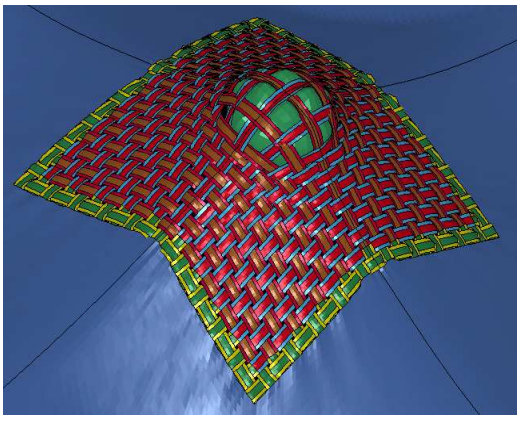
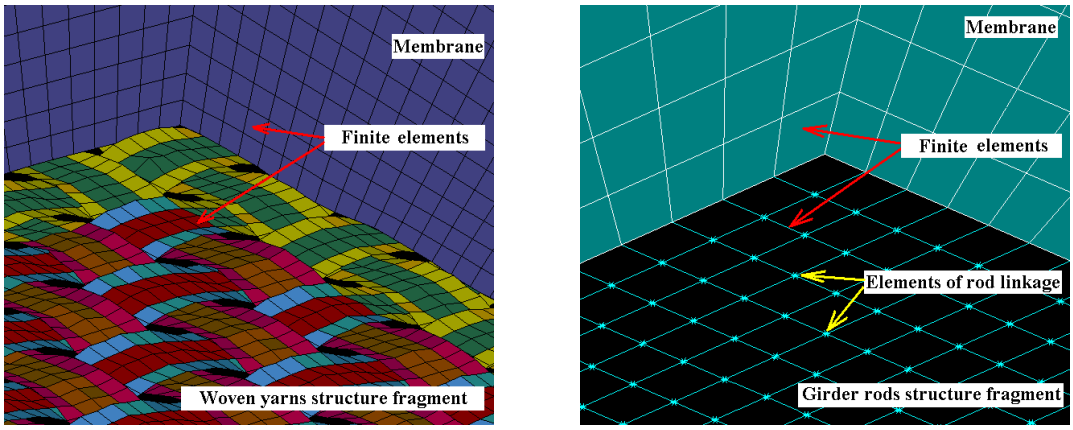


Figure 2. Reference model when a bullet does not penetrate textile target

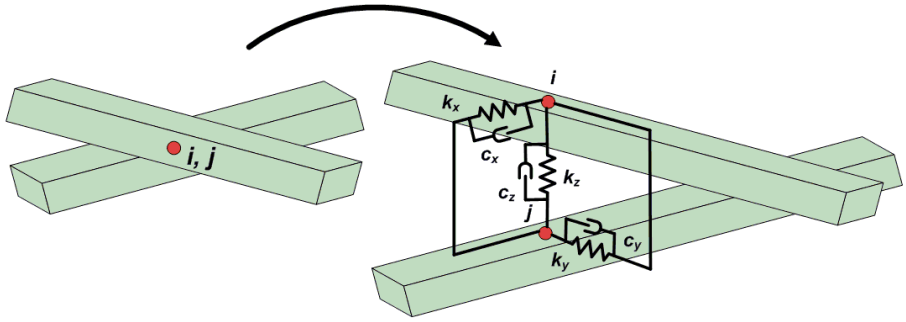
Reduction of finite element models significantly reduces calculation scope while maintaining practically acceptable modelling accuracy. Model reduction task is to create geometrically non-linear dynamic FE reduced model whose detail reference model is a complex finite elements computational structure for ballistic contact interaction of a bullet against a textile structure implemented in LS-DYNA environment. Since textile structure in the reference model is rather complex, we will replace it by simpler rod structure named as girder model (see Fig.3).



a) **b)**

Figure 3. Reference model (a) and reduced FE model (b)

Girder rods are not woven and interconnected by the visco-elastic links at intersection points (node - i and node - j) (see Fig.4). In such way a woven textile is simulated.



a) **b)**

Figure 4. Girder rods (a) and their linkage at intersection point (b)

Link of the rods at their intersection point can be described as follows:

$$\begin{bmatrix} c_x & 0 & 0 & -c_x & 0 & 0 \\ 0 & c_y & 0 & 0 & -c_y & 0 \\ 0 & 0 & c_z & 0 & 0 & -c_z \\ -c_x & 0 & 0 & c_x & 0 & 0 \\ 0 & -c_y & 0 & 0 & c_y & 0 \\ 0 & 0 & -c_z & 0 & 0 & c_z \end{bmatrix} \begin{pmatrix} \dot{u}_i \\ \dot{v}_i \\ \dot{w}_i \\ \dot{u}_j \\ \dot{v}_j \\ \dot{w}_j \end{pmatrix} + \begin{bmatrix} k_x & 0 & 0 & -k_x & 0 & 0 \\ 0 & k_y & 0 & 0 & -k_y & 0 \\ 0 & 0 & k_z & 0 & 0 & -k_z \\ -k_x & 0 & 0 & k_x & 0 & 0 \\ 0 & -k_y & 0 & 0 & k_y & 0 \\ 0 & 0 & -k_z & 0 & 0 & k_z \end{bmatrix} \begin{pmatrix} u_i \\ v_i \\ w_i \\ u_j \\ v_j \\ w_j \end{pmatrix} = \begin{pmatrix} R_{xi} \\ R_{yi} \\ R_{zi} \\ R_{xj} \\ R_{yj} \\ R_{zj} \end{pmatrix} \quad (1)$$

where k_x - rigidity element in the direction x, k_y - rigidity element in the direction y, k_z - rigidity element in the direction z; c_x - viscous friction of the element in the direction x, c_y - viscous friction of the element in the direction y, c_z - viscous friction of the element in the direction z; displacements of i node: u_i - in the direction x, v_i - in the direction y, w_i - in the direction z; R_{xi} - i node's inter-element interaction force in the direction x, R_{yi} - i node's inter-element interaction force in the direction y, R_{zi} - i node's inter-element interaction force in the direction z; and respectively designated displacements of the node j and interaction forces R_{xj}, R_{yj}, R_{zj}.

The girder (textile) areas distant from the interaction area were simulated using membrane elements and larger elements than elements of girder rods (yarns) mesh.

Membrane type environment of uniformly woven textile patch can not represent dynamic behaviour identical to woven textile structure. Nevertheless sufficient similarity is possible provided certain geometrical and physical properties of membrane area are selected. Matching criteria between response of reference and reduced FE models is assumed to be a coincidence of displacements of appropriate nodes – objective function at the same mechanical impacts.

The task is solved in presence of dynamic load. Optimization variables are considered to be the link elements of interlinked rods simulating the textile of reduced FE model. As a measure of quality of the approximation of the reference FE model by reduced FE membrane model, we employ the minimum of a penalty-type objective function expressed as a sum of squares of differences between the displacements of corresponding nodes of each model. The dynamic behaviour of the two structures has been analyzed. In the case of dynamic analysis, the differences between displacements of nodes are minimized at selected time moments. The analysis has been performed by using LS-DYNA, ANSYS and MATLAB software. The displacements obtained in ANSYS have been used when forming the objective function, which subsequently has been minimized by employing MATLAB function FMINCON().

3 Computational results

The graphical illustration of reference model is presented in Figure 5a. Yarn in reference model is composed of finite elements, therefore mesh of elements in the intersection area textile and membrane is 4 times denser comparing to the yarns of their own. Since the construction is symmetrical, only its quarter is simulated, by fixing it between symmetry planes.

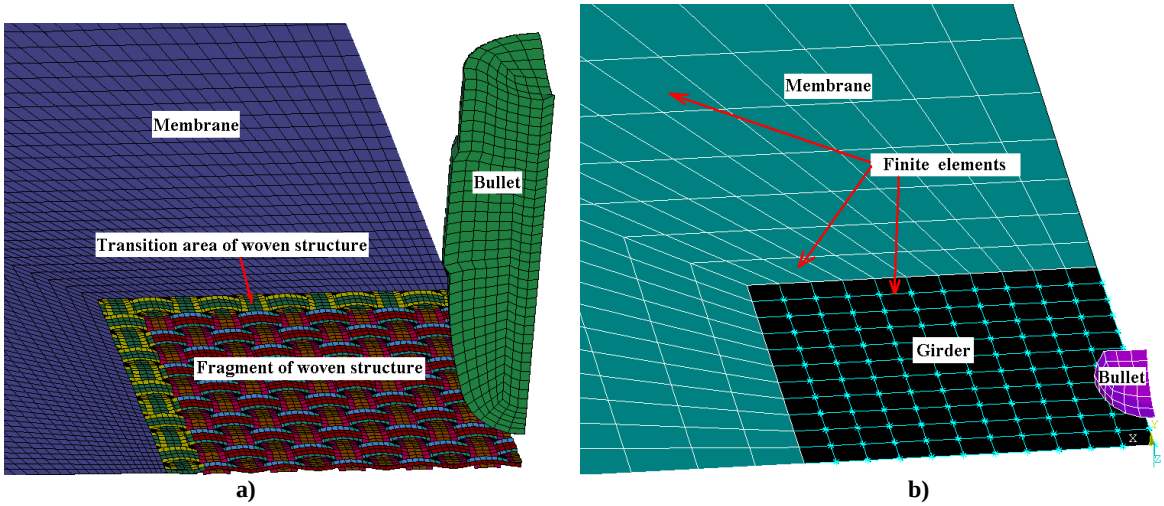


Figure 5. Reference model (a) and reduced FE model (b)

Reduced FE model is composed where a textile yarn is a finite element of a rod, and number of finite elements at the intersection area is the same as number of yarns. Three-dimensional finite element a rod is implemented by applying ANSYS software library element BEAM4. Rods are not woven in FE model, and they are connected by COMBIN14 link elements at intersection points in the directions x, y, z. The graphical illustration of reduced FE model is presented in Figure 5b.

Taking into account quarter symmetry of reference model, we compose rod girder structure 11x11 and surround the membrane which is implemented by elements SHELL63. Mesh of membrane elements is selected so that finite element increases with a distance from the girder, which results in a lower number of finite elements. CONTA175 and TARGE170 elements were applied for description of textile contacts against bullet. Bullet in the software ANSYS is implemented by elements of type SHELL63.

Membrane thickness can be chosen freely, but Young's module E, material mass density ρ, and shear module of orthotropic material G must be chosen thoroughly to obtain constructions' dynamic characteristics closer to each other. We create reduced FE model with the same physical parameters as the reference model. We chose measuring system (kg-mm-s) kilogram-millimetre-second and define physical parameters of the constructions.

Reference textile target. Cross-section of the yarn becomes not circular but elliptical after yarns are woven in the textile. Therefore yarn's thickness at the centre is $1.5 \cdot 10^{-1} mm$, and yarn thickness at the edges is

$7.5 \cdot 10^{-2} mm$. Yarn material mass density is $1400 \cdot 10^{-9} \frac{kg}{mm^3}$, Young's module is $9.0 \cdot 10^{7} Pa$. Yarn's thickness is a half of width at the x and y axes of symmetry.

Reference textile's target transition area. Thickness at the yarn's centre is $1.5 \cdot 10^{-1} mm$, and yarn's thickness at the edges is $7.5 \cdot 10^{-2} mm$. Yarn's material mass density is $1000 \cdot 10^{-9} \frac{kg}{mm^3}$, Young's module is $9.0 \cdot 10^{7} Pa$.

Reference textile's membrane. Membrane thickness $1 \cdot 10^{-1} mm$, material mass density $2250 \cdot 10^{-9} \frac{kg}{mm^3}$, Young's module $2.75 \cdot 10^{7} Pa$, shear module is $2.0 \cdot 10^{5} Pa$, Poisson ratio is equal to 0, number of elements along one side is 44.

Girder of a reduced FE model. Rod thickness $1.125 \cdot 10^{-1} mm$ (average of thicknesses of the reference yarn centre and of the edge), rod width $7.8 \cdot 10^{-1} mm$ (measured in the reference model), rod's material mass density is $1400 \cdot 10^{-9} \frac{kg}{mm^3}$, Young's module is $9.0 \cdot 10^{7} Pa$. Rod's width at the x and y axes of symmetry is $3.9 \cdot 10^{-1} mm$, because we are modelling a quarter of the construction.

Membrane of reduced FE model. Thickness is $1 \cdot 10^{-1} m$, material mass density is $2250 \cdot 10^{-9} \frac{kg}{mm^3}$, Young's module is $2.75 \cdot 10^{7} Pa$, shear module is $2.0 \cdot 10^{5} Pa$, Poisson ratio is equal to 0, number of elements along one side $N_m = 11$, viscosity coefficient is $1 \cdot 10^{-6} Pa \cdot s$.

Matching criteria between response of reference and reduced FE models is assumed to be a coincidence of displacements of appropriate nodes – objective function at the same ballistic impact. Dynamics and time of bullet travel is known.

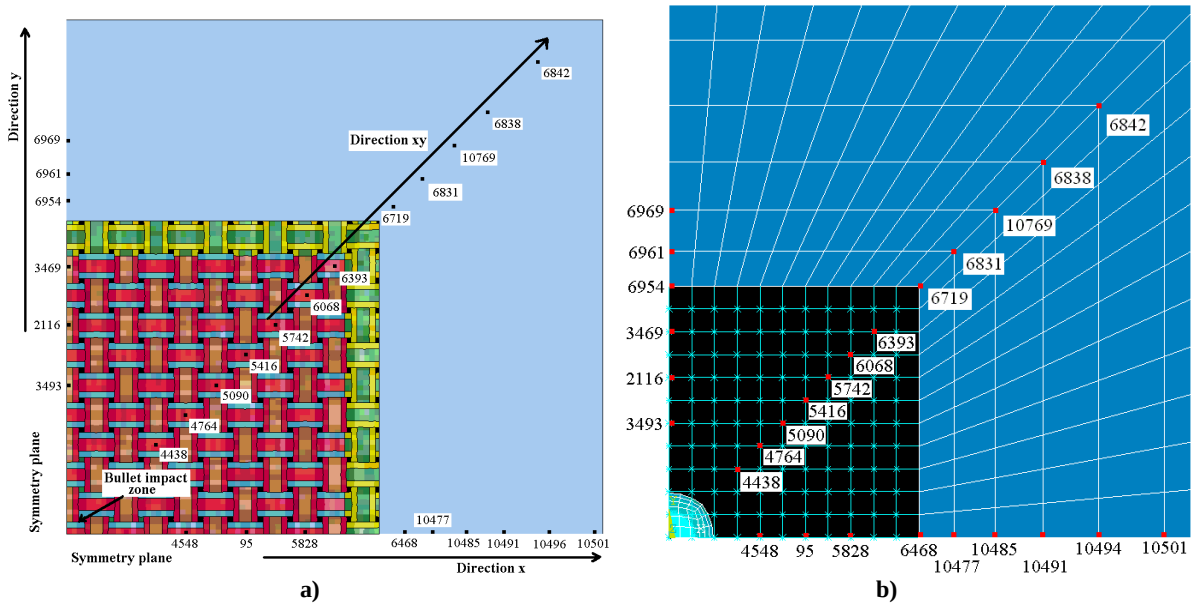
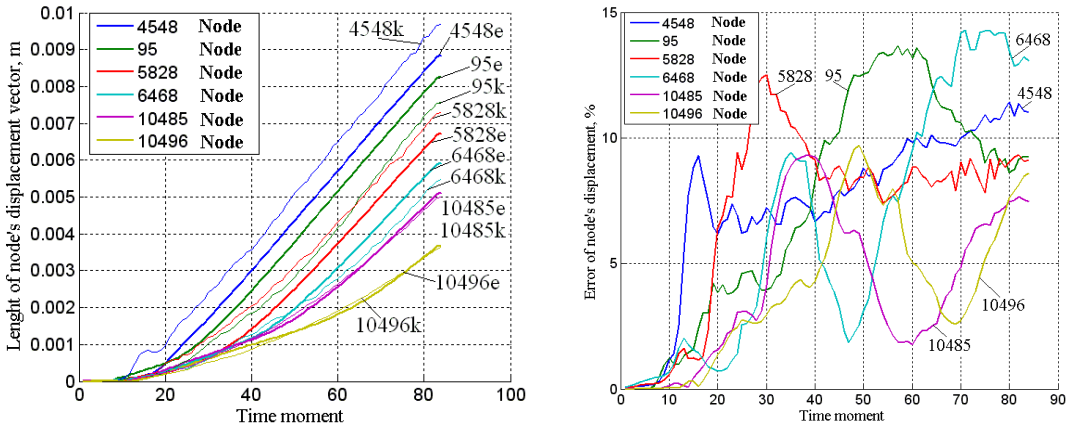


Figure 6. Reference model (a) and reduced FE model (b) with indicated characteristic nodes

We ***empirically*** chose rigidity of link elements at connection points of reduced FE model girder rods, i.e. in x and y directions: $k_x = k_y = 3.8661 \cdot 10^{6} \frac{N}{mm}$, and in z direction: $k_z = 5.2448 \cdot 10^{3} \frac{N}{mm}$.

Figure 7 and Figure 8 show displacements and errors of characteristic nodes in the case of empiric selection of reduced FE model's physical parameters comparing to reference model's characteristic nodes. Numbers of characteristic nodes are indicated in the figures. Numbers are accompanied by the symbols „k" and „e", which mean: „k"- node belongs to reduced FE model, „e" – node belongs to reference model.

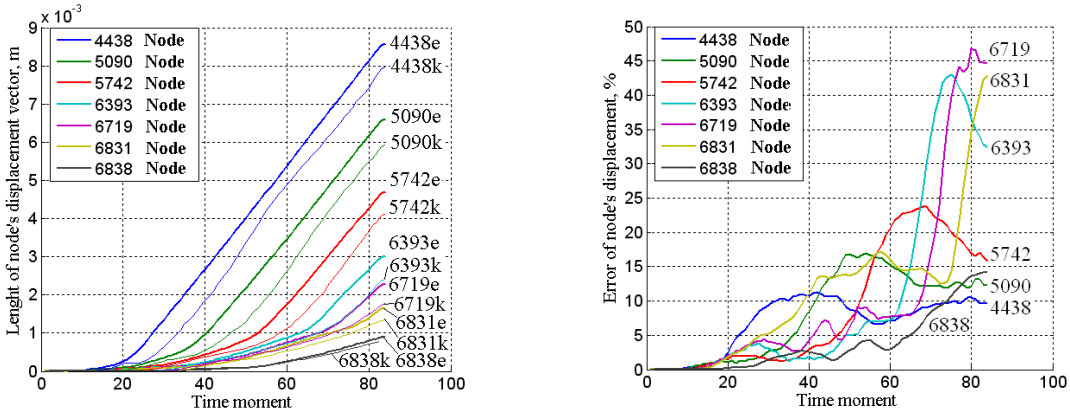


a) b)

Figure 7. Displacements (a) of model characteristic nodes on the x axis and their errors (b)

From the Figure 7a, we can see how characteristic nodes move in both FE models, and maximal error value is approximately 15% (see Fig.7b). Higher error values are for characteristic nodes 95 and 6468.

Below, displacements and errors of characteristic nodes of reference and reduced FE models on xy axis are compared. Diagrams are presented in Figure 8.



a) b)

Figure 8. Displacements (a) of model characteristic nodes on xy axis and their errors (b)

From the Figure 8a, we can see how characteristic nodes move in both models, and maximal error value is approximately 47% (see Fig.8b). Maximal error values are for the nodes 6393, 6831 and 6719 of the reduced FE model.

Optimization task is solved in order to ***adjust*** physical parameters of the reduced FE model. We will apply MATLAB software's optimization function FMINCOM(). Objective function is defined as follows:

$$T(\vec{p},\vec{q}) = \frac{\sum_{i=1}^{n}(\vec{p}_i - \vec{q}_i)^2}{\sum_{i=1}^{n}(\vec{p}_i)^2 + \sum_{i=1}^{n}(\vec{q}_i)^2}, \tag{2}$$

where $\vec{p}_i$ is the vector of i-node displacements of the reference model, $\vec{q}_i$ is the vector of i-node displacements of the reduced model, n - total number of the nodes of each model.

As optimization variables, we use link elements of interconnected rods simulating textile target of the reduced FE model: k_x - in the direction x, k_y - in the direction y, k_z - in the direction z. Since reduced FE model is symmetrical then $k_x = k_y$. Following limitations are determined experimentally for the optimization task:

$$\begin{cases} 1.5\cdot 10^3 < k_x < 5.0\cdot 10^4, \\ 4.5\cdot 10^5 < k_z < 7.0\cdot 10^6. \end{cases} \tag{3}$$

Solution of optimization task showed that rigidity of link elements at connection points of the rods in the directions x and y is $k_x = k_y = 3.8661 \cdot 10^3 \frac{N}{mm}$, and in the direction z - $k_z = 5.2448 \cdot 10^6 \frac{N}{mm}$.

Below, comparison of reduced FE model (see Fig.9) against reference computational model is presented (see Fig.2).

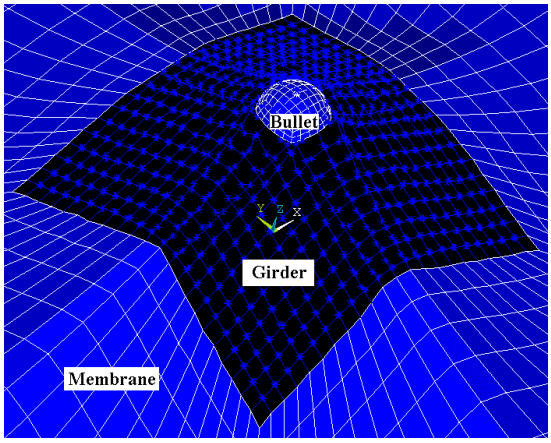


Figure 9. Reduced FE model at the time moment $t = 5 \cdot 10^{-5} s$

Maximal error between characteristic nodes of reference and reduced FE models was 14% after adjustment of physical parameters. Matching criteria between responses of both models is assumed to be coincidences of displacements of characteristic nodes, diagrams for which are presented in Figure 10 and Figure 11.

Displacements of characteristic nodes of reference and reduced FE models on x axis and their errors are presented in Figure 10.

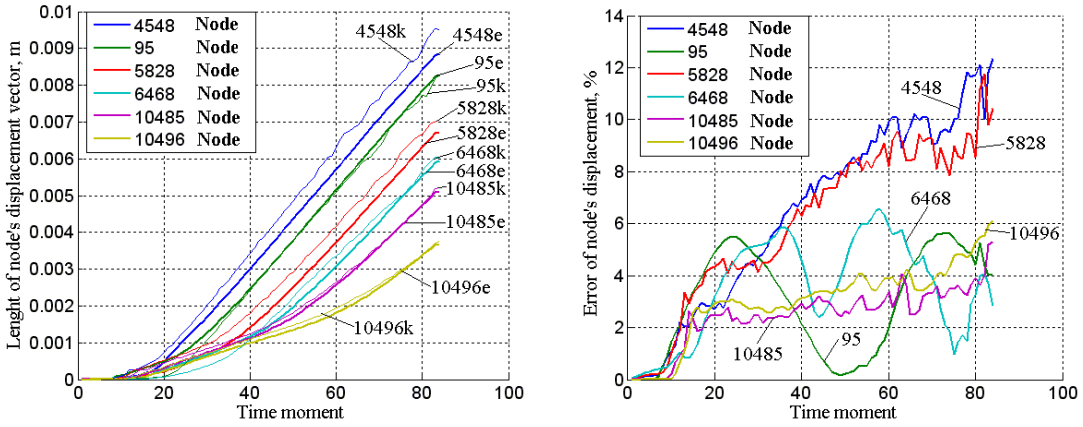


Figure 10. Displacements (a) of model characteristic nodes on the x **axis and their errors (b)**

From the Figure 10a, we can see that characteristic nodes in both models move similarly and maximal error value is 12%. Higher error values are for the nodes 4548 and 5828 (see Fig.10b). Higher errors for the node 4548 could be caused by a small distance from the bullet impact area.

Also characteristic nodes on axis xy of reference and reduced FE models were mutually compared. Diagrams are presented in Figure 11.

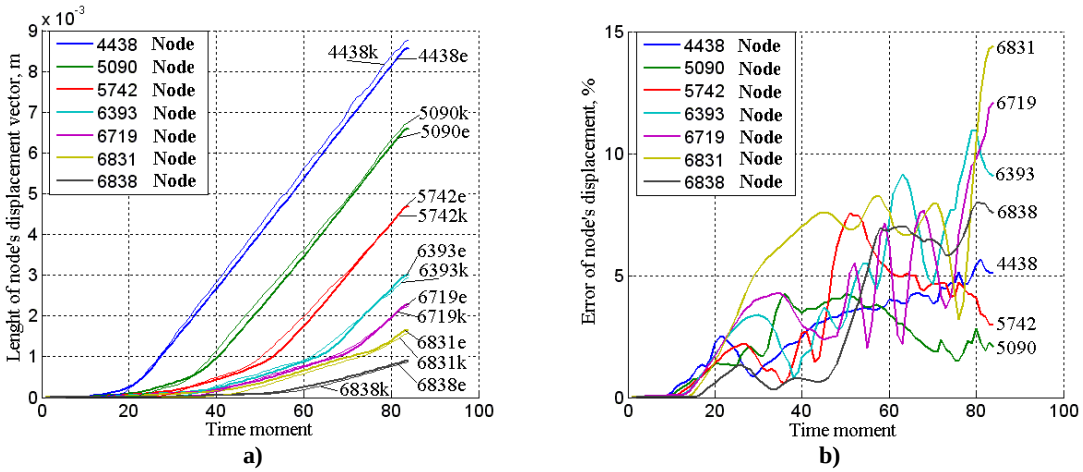


Figure 11. Displacements (a) of model characteristic nodes on the *xy* axis and their errors (b)

From the Figure 11a, we can see that characteristic nodes in both models move similarly and their errors are presented in Figure 11b. In this case, maximal error value is about 3.5 times lower than maximal error determined in Figure 8b.

4 Conclusions

A formal approach to the reduction of a complex finite element structural model to the simpler one has been proposed. The procedure is based on penalty type objective function minimization in the space of parameters of the reduced model. As a sample task, the synthesis of the reduced finite element model, which imitates the behaviour of the reference structure, has been solved.

A reduced finite element model has been developed, the reference model of which was a complex ballistic contact interaction model of a bullet against a textile structure implemented in LS-DYNA finite element system. The structure of the reduced model was formed of elastic beam elements, connected by visco-elastic link elements. The constitution of the obtained model is well simpler, therefore could be calculated several times quicker. Meanwhile, the response of the reduced model was up to 10-14% different compared to the response of the reference model.

References

[1] **Barauskas R.** Modeling of the bullet perforation of textile targets by using combined woven structure – membrane approach. *WSEAS Transactions on Information Science and Applications*, November 2005, Volume 2, Issue 11 1944-1954, ISSN 1790-0832.

[2] **Barauskas R., Abraitiene A., Vilkauskas A.** Simulation of a ballistic impact of a deformable bullet upon a multilayer fabric package. *2nd International Conference on Computational Ballistics, WIT Press, Southmpton, Boston*, 2005, 41-51.

[3] **Barauskas R., Abraitiene A.** Computational analysis of impact of a bullet against the multilayer fabrics in LSDYNA. *International Journal of Impact Engineering*, July 2007, Volume 34, Issue 7, 1286-1305.

[4] **Barauskas R., Kuprys M.** Collision detection and response of yarns in computational models of woven structures. *Proceedings of 10th International Conference Mathematical Modeling and Analysis and 2nd International Conference Computational Methods in Applied Mathematics*, June 1-5, 2005, Trakai, Lithuania, 1-6.

[5] **Clegg R., Hayhurst C., Leahy J., Deutekom M.** Application of a coupled anisotropic material model to high velocity impact response of composite textile armour. *18th International Symposium and Exhibition on Ballistics, San Antonio, Texas USA*, November 15-19, 1999, 791-798.

ADAPTIVE MOVING ALGORITHM OF MOBILE PIEZOROBOT

Asta Drukteiniene[1], Ramutis Bansevicius[2], Genadijus Kulvietis[1]

[1] *Vilnius Gediminas Technical University, Sauletekio 11, LT-10223, Vilnius, Lithuania, astad@it.su.lt, Genadijus_Kulvietis@gama.vtu.lt*

[2] *Kaunas University of Technology, Kestucio 27, LT-44312, Kaunas, Lithuania, bansevicius@cr.ktu.lt*

Abstract. Analysis of rotating mobile piezorobot movement algorithm showed that deflection from given movement function increases with increasing moving step. This paper presents adaptive motion method of mobile piezorobot. Preliminary experimental results prove the feasibility of proposed mathematical model.

Keywords: mobile piezorobot, moving algorithm, adaptive trajectory planning method.

1 Introduction

Mobile piezorobots are devices capable of manipulating objects of small mass in a limited space but with very high accuracy (0.1 μm) [3-6]. They are designed from piezoceramic plate with three magnetized metallic cylinders that are attached to piezoceramic plate.

Mobile piezorobots by form can be divided into: piezoconverter ring, piezoconverter cylinder and piezoconverter shell. Electrodes cover all bottom space of the piezoconverter and are divided into three equal segments. Such electrodes division allows excite slider move in any direction and rotary motion. Electrodes can be divided into a greater number of sectors, but exciting principle should remain the same.

Movement of this kind of robot can be described in two methods: switching leg method, when robot can't rotate [1], and rotating piezorobot moving method [2].

General requirements for piezorobot movement are:

1. Move by given function $\psi = f(x, y)$. Function ψ must be continuous at each point and its derivatives at those points must be continuous too.
2. Deflection ε must be minimum from function ψ.

Analysis of the results of the rotating mobile piezorobot movement algorithm showed, that function vertexes are cutting away, when moving step is increasing. Step value is assigned at startup and is constant. Moving trajectory is distorting and not exactly matching given function at its vertexes. Thus, with this method, when radius of curvature is decreasing, deflection from given function is increasing. So it is necessary to determinate maximum deflection, which cannot exceed the mobile piezorobot center. There can be adjusting switching leg method used.

This paper presents adaptive motion algorithm of rotating mobile piezorobot.

2 Adaptive Moving Algorithm

2.1 Trajectory planning method

This data must be determinated in order to describe rotating mobile piezorobot motion:

c – amount of power actuators;

α_0 – angle between first power actuator and x axis;

Δs_0 – maximum step of mobile piezorobot;

D – general direction of movement with regard to x axis; $D = \text{sign}(x - x_0)$, where x_0 – start coordinate, x – goal coordinate.

Two variables for adaptive moving algorithm must be additionally determinated:

Δs_{min} – minimum step of mobile piezorobot ($\Delta s_{min} < \Delta s_0$);

ε – maximum deflection from given function.

So switching leg method can be adjusted for finding marginal coordinates g^* and g^{**} ($g^* < \psi < g^{**}$):

$$g_i(x_i, y_i) = \begin{cases} x_{gi} = -K \cdot \dfrac{\varepsilon \cdot \left.\dfrac{\partial f}{\partial x}\right|_{x=x_{fi}}}{\sqrt{\left(\left.\dfrac{\partial f}{\partial x}\right|_{x=x_{fi}}\right)^2 + 1}} + x_{fi}; \\ y_{gi} = K \cdot \dfrac{\varepsilon}{\sqrt{\left(\left.\dfrac{\partial f}{\partial x}\right|_{x=x_{fi}}\right)^2 + 1}} + y_{fi}, \end{cases} \quad (1)$$

where K sign.

$$g_i(x_i, y_i) = \begin{cases} g^*(x_i, y_i), & \text{if } K = 1; \\ g^{**}(x_i, y_i), & \text{if } K = -1. \end{cases} \quad (2)$$

Rotating method is base of adaptive moving algorithm, but with accuracy constraint (Figure 1).

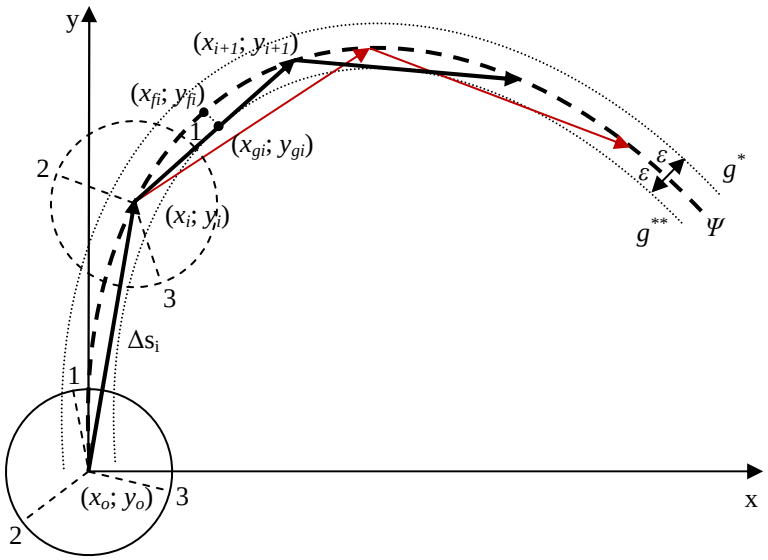


Figure 1. Adaptive mobile piezorobot movement trajectory (black line) and non-adaptive trajectory (red line)

After solving system of equations (3) received results are all possible moving points.

$$\begin{cases} (x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2 = \Delta s^2; \\ f(x_i, y_i) = 0. \end{cases} \quad (3)$$

Only those points which direction matches general moving direction D are selected. Then, verifying piezorobot movement straight line intersect marginal coordinates g^* and g^{**} and how much intersect points are with each function g^* and g^{**} (Figure 2).

For finding all possible intersection points system of equations is solved with each function g^* and g^{**}:

$$\begin{cases} y_{i+1} - y_i = \tan\varphi_i (x_{i+1} - x_i), \\ \begin{pmatrix} x_{i+1} \\ y_{i+1} \end{pmatrix} = K \cdot g_i(x_i, y_i), \\ f(x_{fi}, y_{fi}) = 0, \end{cases} \quad (4)$$

where $(x_{i+1}; y_{i+1})$ – intersect points, $(x_i; y_i)$ – piezorobot center coordinates, $\varphi_i = \operatorname{ctg}\left(\dfrac{y_{i+1} - y_i}{x_{i+1} - x_i}\right)$ angle between moving line and x axis, i = 1, 2 ...n point number.

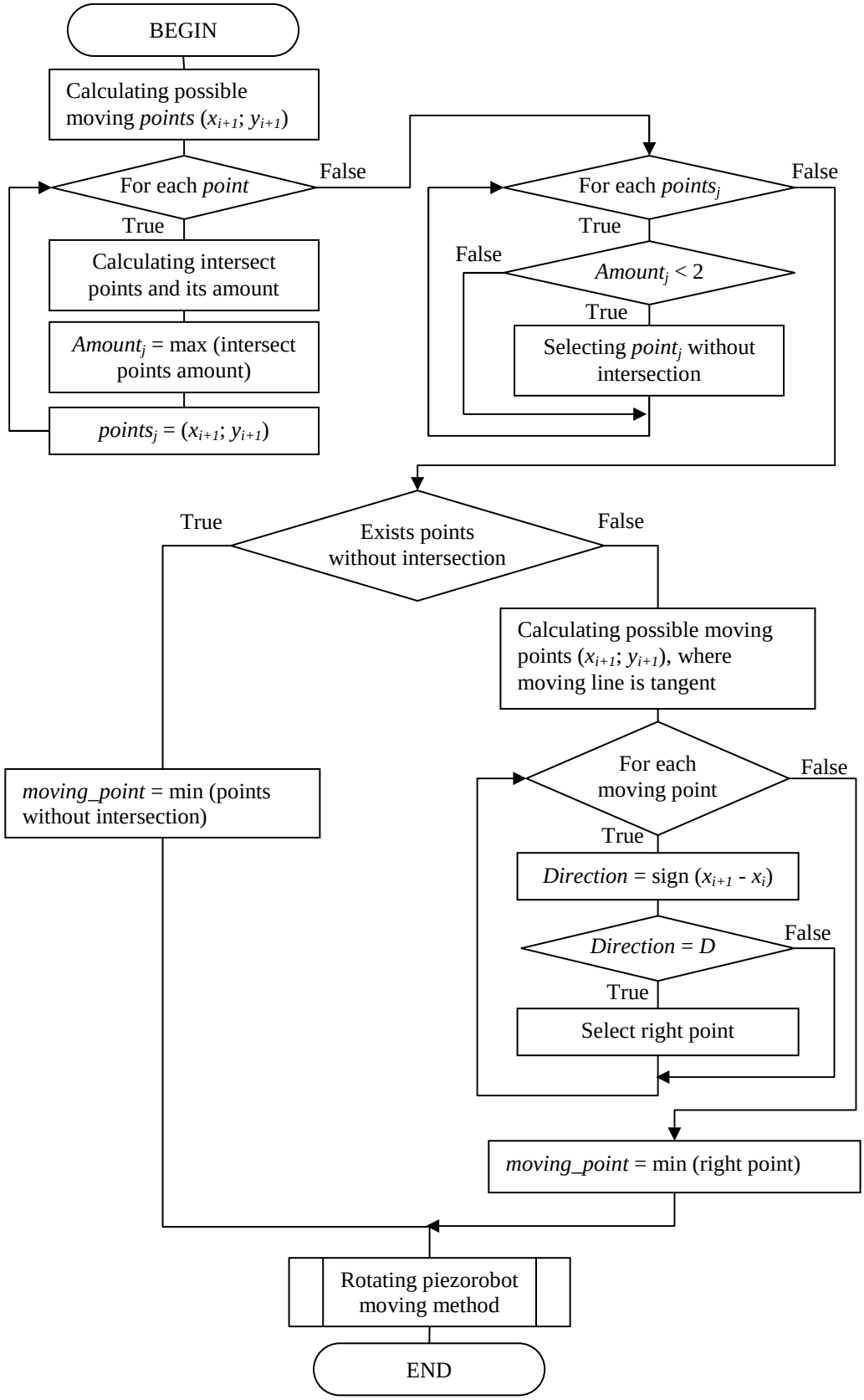


Figure 2. Adaptive motion algorithm

Intersection points amount is calculated for each moving line. If value transcends 1, then points are not valid, otherwise – point is motion coordinate. If there are several right points, then point which is near mobile piezorobot position is chosen.

If there are no moving points, then a point $(x_{i+1};y_{i+1})$ must be found at function ψ, where straight line between points $(x_{i+1};y_{i+1})$ and $(x_i;y_i)$ will be tangent at point (x_{gi}; y_{gi}) (Figure 1).

For this, system of equations is solved:

$$\begin{cases} \dfrac{x_{i+1}-x_i}{x_{gi}-x_i}=\dfrac{y_{i+1}-y_i}{y_{gi}-y_i}, \\ \begin{pmatrix} x_{gi} \\ y_{gi} \end{pmatrix}=K\cdot g_i(x_i,y_i), \\ y_{gi}-y_i=\left.\dfrac{\partial f}{\partial x}\right|_{x=x_{fi}}(x_{gi}-x_i). \end{cases} \tag{5}$$

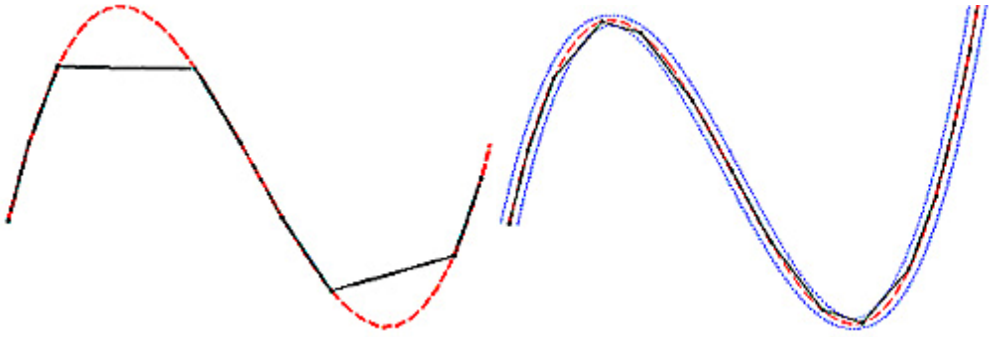
where first equation is straight line between three points equation, second is marginal coordinates and third is tangent equation at point x_{fi}.

Not all points are right for motion, so they are chosen depending on general direction D of movement with regard to x axis and which is near mobile piezorobot position

And finally, by rotating piezorobot moving method mobile piezorobot rotation angle and which power actuator must be active to move on can be found. Distance between two points $(x_{i+1};y_{i+1})$ and $(x_i;y_i)$ is calculated. If this distance is less than minimum piezorobot step Δs_{min}, then robot can't move anymore and stops.

2.2 Results of Analysis

Mobile piezorobot with three power actuators (c = 3) moving trajectory by adaptive moving algorithm is analyzed. Non-adaptive trajectory, when function $y=\frac{1}{2}(x+4)(x+1)(x-1)$, α = 30, Δs_0 = 1.5, Δs_{min} = 0.01, ε = 0.1 is presented in Figure 3a. Adaptive trajectory, with the same values is presented in Figure 3b. Results of computation are presented in Table 1.



a) **b)**

Figure 3. Moving trajectory, when $y=\frac{1}{2}(x+4)(x+1)(x-1)$: a) non-adaptive; b) adaptive

Table 1. Adaptive moving algorithm results

i	j	Rotation direction	$\theta_{min,i}$, °	Δs_i	x_i	y_i
1	0	0	0	0	-4	0
2	1, 2, 3	counterclockwise	51,423	1,346251	-4	0
3	1	0	0	1,5	-3,77629	1,483224
4	1, 2, 3	clockwise	3,465373	0,090723	-3,77629	1,483224
5	1	0	0	1,5	-3,46334	2,950215
6	1, 2, 3	clockwise	14,65467	0,383658	-3,46334	2,950215
7	1	0	0	1,261501	-2,89658	4,077233
8	1, 2, 3	counterclockwise	30,13079	0,788822	-2,89658	4,077233

i	j	Rotation direction	$\theta_{min,i}$, °	Δs_i	x_i	y_i
9	3	0	0	0,529862	-2,42266	3,840262
10	1, 2, 3	clockwise	39,05545	1,022469	-2,42266	3,840262
11	3	0	0	1,5	-1,80352	2,474002
12	1, 2, 3	clockwise	6,254885	0,163753	-1,80352	2,474002
13	3	0	0	1,5	-1,33693	1,048419
14	1, 2, 3	clockwise	0,029736	0,000778	-1,33693	1,048419
15	3	0	0	1,5	-0,87107	-0,37741
16	1, 2, 3	counterclockwise	6,029067	0,157841	-0,87107	-0,37741
17	3	0	0	1,5	-0,25803	-1,74641
18	1, 2, 3	counterclockwise	37,38803	0,978816	-0,25803	-1,74641
19	3	0	0	0,5462	0,222028	-2,00695
20	1, 2, 3	clockwise	28,89755	0,756536	0,222028	-2,00695
21	1	0	0	1,221103	0,78373	-0,9227
22	1, 2, 3	counterclockwise	15,16164	0,396931	0,78373	-0,9227
23	1	0	0	1,5	1,10136	0,54328
24	1, 2, 3	counterclockwise	3,576151	0,093623	1,10136	0,54328
25	1	0	0	1,5	1,326931	2,026222
26	1, 2, 3	counterclockwise	1,688263	0,044199	1,326931	2,026222
27	1	0	0	1,5	1,508715	3,515166

Adaptive moving trajectory with $y = \frac{1}{2}(x+4)(x+1)(x-1)(x-3)$ is demonstrated in Figure 4.

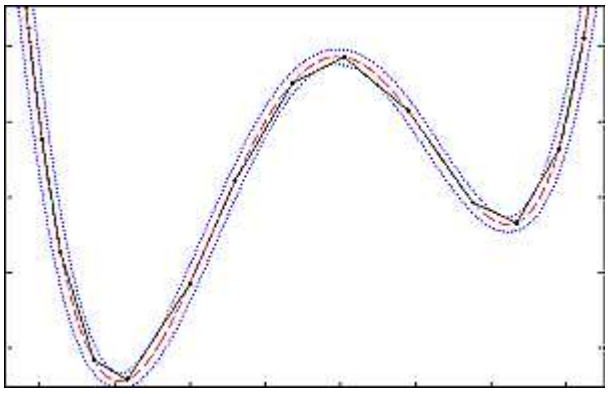


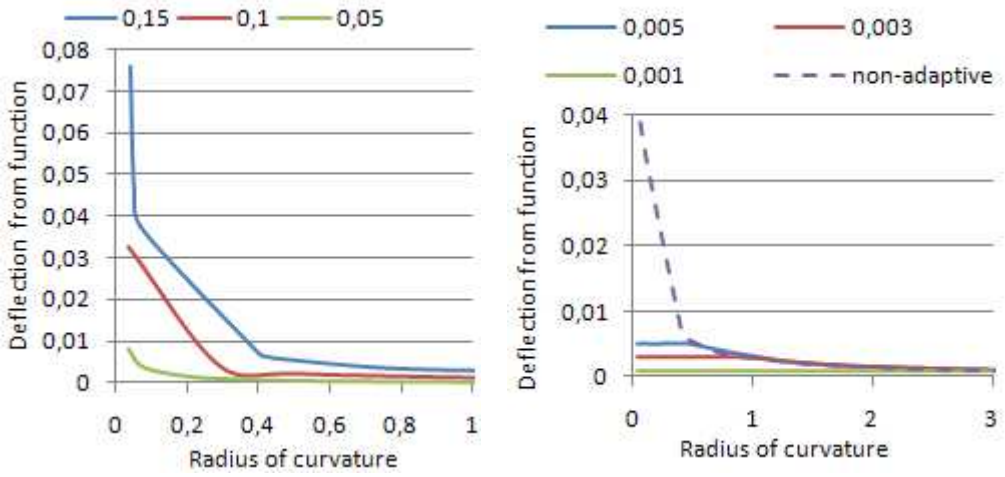
Figure 4. Adaptive moving trajectory

Deflection from function dependence on radius of curvature, when maximum step of mobile piezorobot is changing Δs_0 = {0.15; 0.1; 0.05} is analyzed (Figure 5a). Minimum and maximum deflection from function is calculated with non-adaptive algorithm for function $y = \frac{1}{2}(x+4)(x+1)(x-1)(x-3)$ (Table 2).

Table 2. Minimum and maximum deflection from function

	Δs_0		
	0,15	0,1	0,5
ε_{min}	$2{,}75\cdot10^{-8}$	$1{,}68\cdot10^{-8}$	$7{,}82\cdot10^{-9}$
ε_{max}	$7{,}62\cdot10^{-2}$	$3{,}28\cdot10^{-2}$	$7{,}06\cdot10^{-3}$

Non-adaptive method is compared with adaptive algorithm, with specified maximum deflection from function ε = {0.005, 0.003, 0.001} and Δs_0 = 0.15 (Figure 5b).

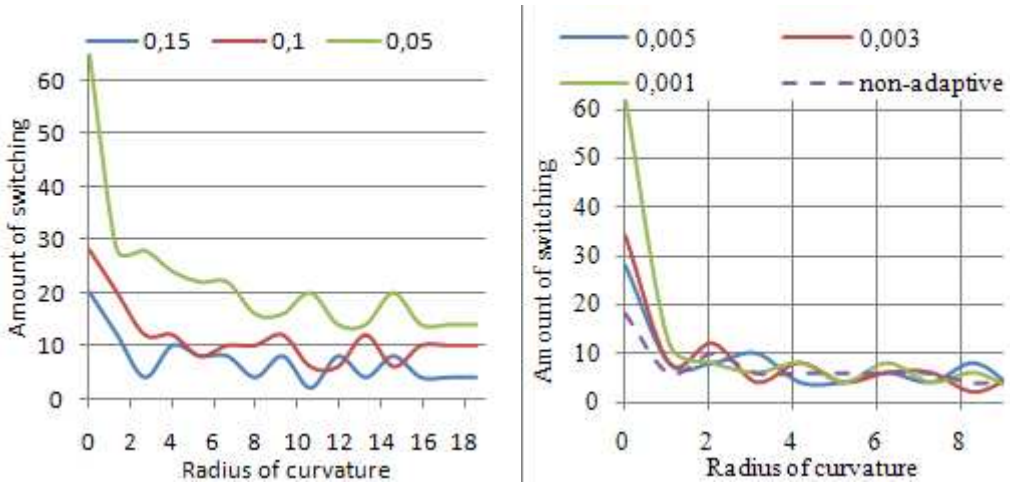


a) **b)**

Figure 5. Deflection from function analysis, when: a) non-adaptive method; b) adaptive method

Analysis results show that with non-adaptive method maximum deflection from function cannot be controlled. With this method only shrink maximum step of mobile piezorobot can be controlled, but deflections are increasing when radius of curvature approximate to 0. With adaptive method deflection from given function can be controlled and it cannot exceed given value.

Switching amount of power actuators dependence on radius of curvature with non adaptive algorithm is analyzed (Figure 6a) too. Maximum step of mobile piezorobot Δs_0 = {0.15; 0.1; 0.05}. These results are compared with non-adaptive moving method, when ε = {0.1, 0.06, 0.02, 0.008} and Δs_0 = 0.15 (Figure 6b).



a) **b)**

Figure 6. Switching amount of power actuators dependence on radius of curvature, when: a) non-adaptive algorithm, b) adaptive algorithm

Analysis results show that amount of switching is increasing when maximum step of mobile piezorobot is decreasing and it significantly increases when radius of curvature approximate to 0. It means that decreasing maximum step of mobile piezorobot, robot will move more slowly and will not be able to develop a maximum speed because of frequently switching contacts and short movement segments.

Using adaptive method for setting the maximum step and the maximum deviation there is possibility to move quickly, because of decreasing need for switch contacts and larger radius of curvature extension of movement line (Figure 7).

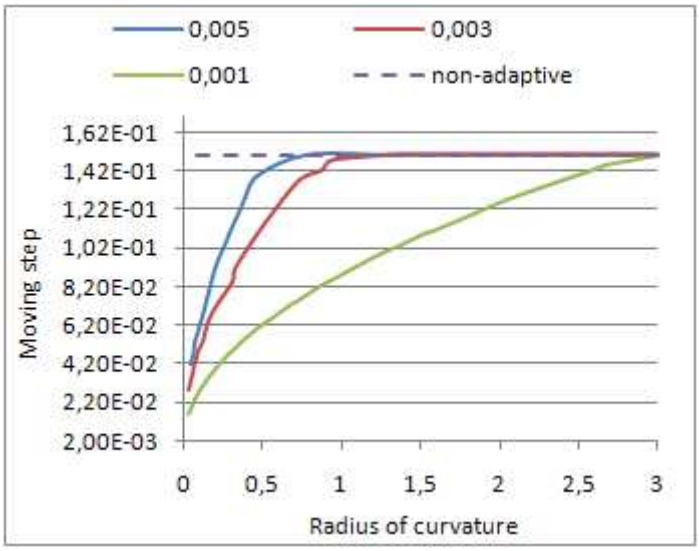


Figure 7. Moving step length dependence on radius of curvature, when ε = {0.1, 0.06, 0.02, 0.008} and Δs_0 = 0.15

But during adaptive method when curvature radius is closing to 0, number of shifts significantly increases. Therefore at the vertexes of function robot will move more slowly, bus more accurately will repeat given function.

3 Conclusions

The adaptive trajectory planning method of mobile piezorobot is presented. The develop method is based on rotating mobile piezorobot method, but with accuracy constraint. Control of the piezorobot is the same as in rotating method.

The presented results prove the feasibility of proposed mathematical model. Results analysis shows, that switching point's amount inversely proportional to radius of given function curvature, but moving trajectory is more close to given function.

4 Acknowledgements

This work has been supported by Lithuanian State Science and Studies Foundation, Project No. B-07017, "PiezoAdapt", Contract No K-B16/2008-1.

References

[1] **Bansevicius R., Drukteiniene A., Kulvietis G**. Trajectory planning method of mobile piezorobot. *Liet. mat. rink. LMD darbai*, 2009, volume 50, 172–177.

[2] **Bansevicius R., Drukteiniene A., Kulvietis G**. Trajectory Planning Method of Rotating Mobile Piezorobot. *J of Vibroengineering, Vibromechanika*, 2009, volume 11 (4), 690-696.

[3] **Juhas L., Vujani A., Adamovi N., Nagy L., Borovac B**. A platform for micropositioning based on piezo legs. *Mechatronics the Science of Intelligent Machines*, 2001, volume 11 (7), 869-897.

[4] **Martel S**. Fundamental Principles and Issues of High-speed Piezoactuated Threelegged Motion for Miniature Robots Designed for Nanometer-scale Operations. *International Journal of Robotics Research* , 2005, volume 24 (7), 575-588.

[5] **Ragulskis K., Bansevicius R., Barauskas R., Kulvietis G.** Vibromotors for Precision Microrobots. *Hemisphere Publishing Corp.*, USA, 1988.

[6] **Uchino K.** Piezoelectric actuators and ultrasonic motors. *Kluwer Academic Publishers*, 1997.

COMPARISON AND TRANSFORMATION OF PIECE LINEAR AGGREGATE TO LINEAR HYBRID AUTOMATON

Arunas Andriulaitis, Vytautas Pilkauskas

Kaunas University of Technology, Department of Business Informatics, Studentu 50-417, LT-51368 Kaunas, Lithuania, a2runas@gmail.com, vytautas.pilkauskas@ktu.lt

Abstract. Many real systems are designed and described by using a mathematical model - piece linear aggregate (PLA). The PLA has tools used to obtain a simulation model and an algorithm used to verify the model. This algorithm is not researched sufficiently. Another mathematical model, linear hybrid automaton (LHA), allows us to verify a model and, in addition, to calculate "safe" values of the model's parameters that satisfy the predefined model's restrictions (i.e. queue cannot be overfilled). We propose the transformation rules from PLA to LHA. After the transformation we can use LHA tools to verify PLA models and calculate the safe values.

Keywords: formal model, transformation, verification of model.

1 Introduction

The piece linear aggregate (PLA) is a formal-mathematical model used to describe and simulate the systems, such as network protocols, complex harbor models, distributed systems, etc [7,9,10]. In order to start verification of the system (verify that the system does not fall into an unexpected state), we need to calculate all reachable states of the system. PLA model has an algorithm used to calculate all reachable states of the system. However, this algorithm is not investigated sufficiently, because there are no methods used to estimate the number of states [9].

Another formal-mathematical model is called linear hybrid automaton (LHA) model. This model does not include the simulation tools to gather statistical data, but LHA model includes the algorithms and tools used to calculate the reachable states of the system (famous examples: steam boiler, power plant, Philips audio system, etc [3,4,5]). Additionally, this model includes the algorithms and tools used to calculate the values of system parameters, ensuring that system will satisfy the predefined invariants (i.e. if parameter's value belongs to the defined interval, the queue would never be overfilled). Parameter's values ensuring that the system never violates the predefined safety conditions (invariants) are called safe values.

In order to verify a system described by using PLA specification and to evaluate the safe values of the parameters, we need to calculate the reachable states. Our main reasons of PLA model transformation into LHA model are as follows: a) to obtain the reachable states by implementing LHA model's algorithms (LHA model includes tools with the algorithms implemented) b) to calculate the safe values of parameters by using tools of LHA model. It is still makes sense to use PLA models as they have developed tools for collecting statistics about model when LHA mainly concentrates only on verification.

Firstly, we describe the PLA and the LHA models. Further, we compare the PLA and the LHA models and define some rules and suggestions regarding the preparation of PLA model for the transformation and then we describe the transformation step-by-step. We provide a short example of the transformation and additional results that we can obtain by using tools of LHA model.

2 Definitions of PLA and LHA models

The PLA model is defined as follows:

$A = (Z, Y, X, E, \Sigma, z(t_0), H)$ [1,2,6],

where:

Z is a set of aggregate states,

$Y = \{y_1, y_2, ...\}$ is a set of outputs from aggregate,

$X = \{x_1, x_2, ...\}$ is a set of inputs to aggregate,

E is a set of events,

Σ is a set of controlling sequences,

H is a set of transition operators,

$z(t_0)$ is an initial aggregate's state during the start-up moment t_0.

The aggregate is analyzed at time moments $t_0, t_1, t_2, \ldots$. Aggregate's state is defined as follows:

$z(t_m) = (\Psi, Z_v) \in Z$

where:

$t_m, m = 0, .., \infty$,

$\Psi = \{\upsilon_1(t_m), \upsilon_2(t_m), ...\}$ is a set of discrete variables,

$Z_v = \{w_1(e_1'', t_m), w_2(e_2'', t_m), ...\}$ is a set of continuous coordinates, where each $w(e'', t_m)$ represents count down clock after which corresponding e'' happens.

The set of the external events is denoted by $E' = \{e_1', e_2', ...\}$, where each event e' occurs when the signal of from corresponding input $x_1, x_2, ...$ arrives. The set of internal events is $E'' = \{e_1'', e_2'', ...\}$, where each event e'' occurs when the corresponding clock $w(e'', t_m)$ reaches 0. Every internal event has a controlling sequence $e_i'' \to \varsigma_i, a_i \le \varsigma_i \le b_i, \varsigma_i \in \Sigma$, where $0 < a_i, b_i \in R$. The length (the clock timer) of the appropriate operation is defined by the controlling sequence.

Transition operators H change aggregate's state, expressed as $z(t_{m+1}) = H(z(t_m), e), e \in E' \bigcup E''$, and then Y set of output signals is produced. Each transition operator has a set of assignments $\phi_{y1}, \phi_{y2}, ..., \phi_{w1}, .\phi_{w2}, ..., \phi_{\upsilon 1}, \phi_{\upsilon 2}, ... \in \Phi$ (new values to variables) dedicated to the outputs *y*, discrete variables υ, as well as continuous variables *w*, which depend on $\varphi_{y1}, \varphi_{y2}, ..., \varphi_{w1}, \varphi_{w2}, ..., \varphi_{\upsilon 1}, \varphi_{\upsilon 2}, ... \in \Gamma$ conditions. One assignment ϕ corresponds to one assignment condition φ.

The LHA model is defined as follows:

$A = (L, V, init, inv, flow, jump, \mathrm{E})$ [3,4,5]

where:

$L = \{l_1, l_2, ...\}$ is a set of continuous variables,

$V = \{v_1, v_2, ...\}$ is a set of control modes (discrete variables),

E is a set of synchronizing events.

The function $flow(v, l)$ defines the first derivate of the continuous variable *l* which is $\dot{l} = dl/dt$ for each control mode ($\dot{l}$ special shorter notation). The function $inv(v, l)$ defines the range of possible *l* values for each control mode v. The function *init* defines the initial control mode and initial values of continuous variables.

The function *jump* defines the transitions of the LHA model from each specific control mode. The *jump* function consists of two parts: a jump condition and a jump assignment. The jump condition $jump\_con(v, \varepsilon, l)$ (for the every control mode *v*) consists of synchronization signal ε (optional) and inequalities with l. The jump assignment $jump\_upt(v, \varepsilon', v', l, l')$ (for the every control mode *v*) is an operator used to assign new values to the continuous variables *l*, define a new control mode v' and issue a synchronization signal ε' (optional) (l' denotes *l* values in the next control mode).

3 Comparison of PLA and LHA

In order to compare the PLA and the LHA models we used a mathematical formalism of timed transition systems [8]. In general, the timed transition systems define model's state, behavior between jumps, jumps, synchronization signals and initial state. We have found that both models (the PLA and the LHA) are similar, because they both have particular initial values defined at the beginning, discrete and continuous variables, range of values and flow (change) functions dedicated for continuous variables, they both preserve values of discrete variables while models vary between jumps, their behavior is synchronized by signals, both models assign new values to discrete and continuous variables during jumps.

With the help of formalism of timed transition systems we have identified a few differences between the PLA and the LHA models:

1. Synchronization signals are used in the PLA model to pass variables' values, whereas that is not possible in the LHA model.

2. Values of every control sequence of PLA model can be generated by using any distribution law. However, the values of continuous variables of LHA model are continuously distributed within the determined range.

3. The LHA has discrete variable "control mode", which controls the flow (*flow*) of continuous variables and the range (*inv*) of values. The PLA does not have variable "control mode".

4. The PLA model has transition operators used for any state of the model, whereas the LHA transition operators are used only for the defined control modes (the values of source and target control modes must be known).

We assume that first two above-mentioned differences will be eliminated by the model's engineer by using signals without values and splitting control sequences. Therefore, further in this paper we investigate only those PLA models that do not have these two differences. We focus only on the last two differences.

In order to eliminate the third difference we propose to add the control mode $u(t_m)$ to the PLA model. The values of this control mode vary in time moments $t_m, m = 0,..,\infty$. However, $u(t_m)$ always is defined by a finite number of values. These $u(t_m)$ values $u_1, u_2, ...$ are estimated by using the following formula:

$$F_u\left(w_1(e_1'', t_{\bar{m}}),..., w_{n_w}(e_{n_w}'', t_{\bar{m}})\right) = \sum_{j=1}^{n_w} \left\langle w_j(e_j'', t_{\bar{m}}) \right\rangle 10^{j-1} \quad (1)$$

where $t_{m-1} < t_{\bar{m}} < t_m$, n_w number of continuous variables.

Values of $\left\langle w_j(e_j'', t_{\bar{m}}) \right\rangle$ are used to show a state of the continuous variable: 0 – passive, 1 – active, 2 – currently activated. In other words, each value of $u(t_m)$ corresponds to a particular combination of the states of continuous variables. By adding the variable $u(t_m)$ to the PLA model, the control mode of PLA model shall semantically become the same as the one of LHA model.

In order to eliminate the fourth difference between the PLA and LHA models we propose to split each transition operator H by using the following sequence of steps:

1. Split each transition operator by removing the conditions φ of assignments and by moving these conditions to the definition of the operator. By implementing this split it shall be ensured that the state of the model would be defined by the known in advance control mode.

2. Split every obtained transition operator by adding the particular value of $u(t_m)$ to the definition of this operator. In this case it shall be ensured that the state of the model would be changed from the known in advance control mode.

After the transformation of the transition operators, the following description of the transition operators form shall be obtained: $H(e, u(t_m), \varphi_u, \varphi_{y1},.., \varphi_{yn_y}, \varphi_{w1},..., \varphi_{wn_w}, \varphi_{\upsilon 1},..., \varphi_{\upsilon n_\upsilon}, \phi_{u(t_{m+1})}) \rightarrow \{\phi_{y1},.., \phi_{yn_y}\}$,

where:

$e \in E' \bigcup E''$,

$\phi_{u(t_{m+1})}$ represents the value assignment to $u(t_{m+1})$, φ_u condition for $u(t_m)$,

the generic expression n_x is used to denote the size of set *x*.

The output signals are defined after the "→" symbol. This is special notation is introduced to make the description of transformation more easily readable.

4 Transformation of PLA to LHA

Further we use the PLA model satisfying the requirements mentioned in Section 3. The proposed transformation rules are not completely new, however, they are improved, comparing to the transformation rules mentioned in [6]. The main difference between the old and new transformation rules lies in the calculation of the values of $u(t_m)$. In old transformation rules the values of discrete variables are included in $u(t_m)$ and are completely eliminated from the model. Whereas in new transformation rules the discrete variables remain the same and the values of discrete variables are not included in $u(t_m)$. From our point of view, this allows us to have a smaller set of u values and, therefore, the transformation becomes easier.

Transformation of PLA model to LHA model is described by the transformation rules when the PLA model is defined as follows:

$X = \{x_i\}, i = 1..n_x$ represents the set of inputs,

$Y = \{y_i\}, i = 1..n_y$ - set of outputs,

$E' = \{e_i'\}, i = 1..n_x$ - set of external events,

$E'' = \{e_i''\}, i = 1..n_w$ - set of internal events,

$\Sigma = \{\varsigma^i\}, i = 1..n_w$ - controlling sequences,

$u(t_m)$ - additional discrete variable – control mode,

$\Psi = \{\upsilon_i(t_m)\}$, $i = 1..n_\upsilon$ - discrete variables,

$Z_v = \{w_i(e_i'', t_m)\}$, $i = 1..n_w$ - continuous coordinates.

$z(t_0)$ represents the initial state.

The transition operators are defined as follows:

$H(e, u(t_m), \varphi_u, \varphi_{y1}, .., \varphi_{yn_y}, \varphi_{\upsilon 1}, ..., \varphi_{\upsilon n_\upsilon}, \phi_{u(t_{m+1})}) \to \{\phi_{y1}, .., \phi_{yn_y}\}$.

The following rules define transformation:

1. The values of the LHA control modes semantically are the same as the ones of the PLA discrete variable $u(t_m)$: $v_i = u_i, i = 1..n_u$.

2. The values of the LHA continuous variables semantically are the same as the ones of the PLA continuous coordinates $l_{wi} \to w_i(e_i'', t_m), i = 1..n_w$ and PLA discrete variables $l_{\upsilon i} \to \upsilon_i(t_m)$, $i = 1..n_\upsilon$.

Every value $u_i, i = 1..n_u$ of the control mode has the defined functions *inv* and *flow*.

3. The range functions *inv* of the LHA variables l_{wj} and l_{vj} are as follows:

a) for l_{wj}, $inv(u_i, l_{wj}) : l_{wj} \in [0, b_j]$ when u_i includes $\langle w_i(e_i'', t_m) \rangle = 1$ (refer to formula 1),

b) for l_{wj}, *inv* is undefined when u_i includes $\langle w_i(e_i'', t_m) \rangle = 0$ (refer to formula 1).

c) for l_{vj}, $inv(u_i, l_{\upsilon j}) : l_{\upsilon j} = l_{\upsilon j}$.

4. The change functions *flow* of the LHA variables l_{wj} and l_{vj} are as follows:

a) for l_{wj}, $flow(u_i, l_{wj}) : \dot{l}_{wj} = 0$ when u_i contains $\langle w_i(e_i'', t_m) \rangle = 0$ (refer to formula 1),

b) for l_{wj}, $flow(u_i, l_j) : \dot{l}_{wj} = 1$ when u_i contains $\langle w_i(e_i'', t_m) \rangle = 1$ (refer to formula 1),

c) for l_{vj}, $flow(u_i, l_{\upsilon j}) : \dot{l}_{\upsilon j} = 0$.

5. The initial state *init* of the LHA model is equal to the initial state $z(t_0)$ of the PLA model. The corresponding state variables of the different models are related to each other as follows:

$l_{\upsilon j} = \upsilon_j(t_0), j = 1..n_\upsilon$,

and $l_{wj} \in [a_j, b_j]$ when $w_j(e_j'', t_0) = \varsigma_j$

or $l_{wj} = \infty$ when $w_j(e_j'', t_0) = \infty$ ($j = 1..n_w$).

$u(t_0)$ defines initial v.

6. The LHA synchronization signals are represented as PLA output signals $\varepsilon_j \to y_i^1, i = 1..n_y$ (*j* represents the global synchronization index for all automata; *i* represents one index of PLA aggregate outputs).

Transition operators *jump_con* and *jump_upt* are transformed for every control mode v_i:

7. The specific PLA transition operator $H(e, u(t_m), \varphi_u, \varphi_{y1}, .., \varphi_{yn_y}, \varphi_{\upsilon 1}, ..., \varphi_{\upsilon n_\upsilon}, \phi_{u(t_{m+1})}) \to \{\phi_{y1}, .., \phi_{yn_y}\}$ is transformed into the appropriate condition function and the transition assignment function of the LHA model.

The function of jump condition $jump\_con(v, \varepsilon, l_i)$ is described as follows:

a) in case *e* represents an external event, then:

- control mode v is defined by $u(t_m)$,
- synchronization signal $\varepsilon \to x_i \to e$,
- *jump_con* is not influenced by l_i.

b) in case *e* represents an internal event e'', then:

- *jump_con* is not influenced by the synchronization signal ε,
- control mode v is defined by $u(t_m)$,
- conditions of the PLA assignment semantically correspond to the ones of the LHA continuous variables.
- an additional condition $l_w = 0$, which is identical to the corresponding PLA condition $w(e'', t_m) = 0$, is included.

Jump assignment $jump\_upt(v_i, \varepsilon, v_j, l_g, l_g^{'})$ is described as follows:

- the LHA synchronization signals match PLA output signals $\{\varepsilon\} \to \{\phi_{y_1}, ..., \phi_{y_{n_y}}\}$.
- $l_{\upsilon_i} = \upsilon_i(t_m)$, $i = 1..n_\upsilon$.
- the assignments of the expression $w(e'', t_{m+1}) = w(e'', t_m)$ are transformed into $l_w^{'} = l_w$.
- the assignments of the expression $w(e'', t_{m+1}) = \varsigma$ are transformed into $l_w^{'} \in [a, b]$.

These transformation rules do not have enough evidences to prove that the PLA model and the LHA model will have the same reachable states after the transformation. However, it was showed that in some examples the reachable states of two models match.

5 An example of transformation and analysis

The following model is analyzed: generator, controller and 2 processors. The generator generates a signal every 1.3 of the time unit. The signal is passed to the controller, which sends it to the first processor (in case it is not busy). When the first processor is busy, the controller sends the signal to the second processor, and, in case even the second processor is busy, the controller stores the signal in a queue. After any processor finishes its task, this processor informs the controller and receives the next signal. The length of the queue is limited to 10 signals. The first processor can finish the task within [1.4-1.9] time units, the second – [1.8-1.9]. It is required to verify whether the queue can be overfilled (the length of the queue must be retained smaller that 10 all the time).

The PLA specification for the first processor is provided below:

1. Inputs.

$X = \{x_1\}, n_x = 1, x_1^* = \{work1\}$, where x_1^* – signal informing the processor to start work.

2. Outputs.

$Y = \{y_1\}, n_y = 1, y_1^* = \{idle1\}$, where y_1^* – signal informing that the processor has finished the task.

3. Set of the events

External events:

$E' = \{e_1'\}$, where e_1' - event, occurring after signal x_1^* is activated.

Internal events:

$E'' = \{e_1''\}$, where $n_w = 1$, e_1'' - event, occurring after the task is finished.

4. Control sequences.

$e_1'' \to \{\varsigma_j^1\}_{j=1}^{\infty}$, where $1.4 \le \varsigma_j^1 \le 1.9$.

5. Set of the discrete variables.

$\Psi = \{\varnothing\}$. $n_\upsilon = 0$.

Set of the continuous coordinates.

$Z_v = \{w_1(e_1'', t_m)\}$.

6. **Initial state.**

$z(t_0) = \{w_1(e_1'', t_0)\} = \{\infty\}$.

7. Transition operators.

$H(e_1')$:

$n_{\phi w_1 e_1'} = 1$

$\phi_{w_1 e_1' 1} = [w_1(e_1', t_{m+1}) = \varsigma^1]$, $\varphi_{w_1 e_1' 1} = [true]$

$H(e_1'')$:

$n_{\phi y_1 e_1''} = 1$

$\phi_{y_1 e_1'' 1} = [y_1 = idle1]$, $\varphi_{y_1 e_1'' 1} = [true]$

After the transformation of the PLA model to the LHA model the appropriate LHA model for the first processor (Figure 1) is obtained.

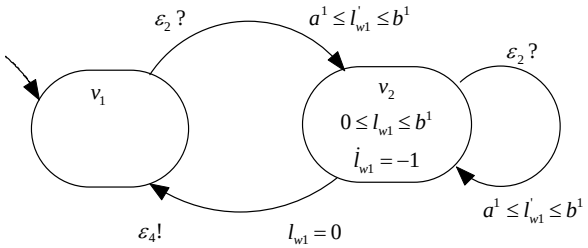


Figure 1. LHA model diagram for the first processor

We have implemented several statistical tests to estimate the average length of the queue (Figure 2). We measured average usage of the first processor and the second processor in relation to the speed of the processors.

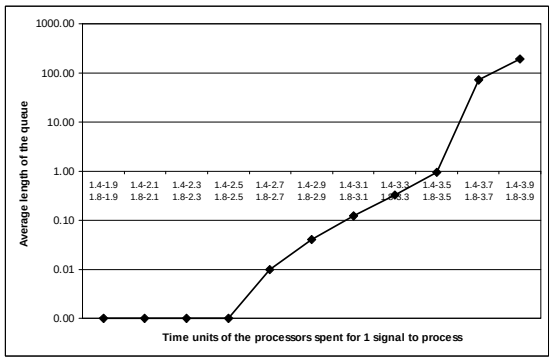


Figure 2. Average length of the queue

According to Figure 2, it is possible to foresee that the model within the given conditions would not reach the state where the queue is overfilled; on the other hand, it is possible to foresee that the queue is overfilled when the first processor spends [1.4-3.6] time units to process a single signal and the second processor spends [1.8-3.6] time units.

Further, we verify the restriction that the number of the signals in a queue is not larger than 10. For this purpose, HyTech software tool (developed by the scientists of Berkeley University and intended for the verification of the small LHA models and calculations of the safe values) is used. HyTech tool confirms that under given initial conditions the restriction is satisfied.

The restriction is violated when the first processor spends [1.4-2.8] time units to process a single signal and the second processor spends [1.8-2.8] time units. Such results show that the statistical data of the PLA model are not accurate enough to verify the restriction in the case where the probability of the violation of the restriction is very low.

Further, every state calculated by HyTech is compared with the manually calculated states of the PLA model. The results show that all the states of HyTech output match the original states of the PLA model. This proves that the preparation and transformation of the PLA model are valid at least for the current example.

Following this comparison, the maximal values of the time units to process a single signal are estimated for both processors. This estimation is implemented by calculating the α for the case where the first processor spends [1.4-1.9] time units for one signal, and the second processor spends $[1.8-\alpha]$. The results provided by HyTech show that $\alpha < 27/10$ for the second processor. Then α is calculated for the case where the first processor spends $[1.4-\alpha]$ time units for one signal and the second processor spends [1.8-2.7). The results provided by HyTech show that $\alpha \le 38/20$ for the first processor. These calculated α values cannot be obtained by using the PLA tools.

HyTech demonstrates the possibilities that are not available in the PLA tools. These possibilities include automatic verification of the model and calculation of the safe values.

6 Conclusions

After our research we can conclude that the comparison of the PLA and the LHA models allows us to define the rules describing the following:

- transformation of the operators;
- calculation of the control modes.

Additionally, we can conclude that after the PLA transformation into the LHA the following can be obtained automatically:

- all possible states of the given model;
- verification of any state of the model;
- safe values of the parameters.

References

[1] **K.Wang, H.Pranevicius**. Application of AI to Production Engineering, *Nordic-Baltic Summer School'97, Kaunas University of Technology, Technologija*, 1997, pp. 275-282.

[2] **H.Pranevicius, V.Pilkauskas, A.Chmieliauskas**. Aggregate approach for specification and analysis of computer network protocols, *Kaunas University of Technology, Technologija*, 1994.

[3] **T.A.Henzinger**. The Theory of Hybrid Automata, *IEEE Computer Society Press*, 1996, pp. 278-292.

[4] **R.Alur, C.Courcoubetis, N.Halbwachs, T.A.Henzinger, P.H.Ho, X.Nicollin, A.Olivero, J.Sifakis, S.Yovine**. The Algorithmic Analysis of Hybrid Systems, *Proceedings of the 11th International Conference on Analysis and Optimization of Systems Discrete Event Systems, Lecture Notes in Control and Information Sciences*, 1994, vol. 199, pp. 329-351.

[5] **Y.Kesten, A.Pnueli, J.Sifakis, S.Yovine**. A New Class of Decidable Hybrid Systems, *Lecture Notes in Computer Science*, 1991, vol. 736, pp. 179-208.

[6] **V.Pilkauskas, A.Andriulaitis**. Transformation of piece linear aggregate to linear hybrid automaton. *Proceedings of the International Conference on Operational Research: Simulation and Optimization in Business and Industry*, Tallin, 2006, pp. 69-74.

[7] **H.Pranevicius, D.Makackas**. The use of aggregate approach for formal specification and simulation of real-time systems, *Databases and Information Systems: Proceedings of the Fourth International Baltic Workshop, Baltic DB&IS 2000*, Vilnius, 2000, pp. 189-198.

[8] **T.Henzinger, Z.Manna, A.Pnueli**. Timed transition systems, *Proceedings of the Real-Time: Theory in Practice, Lecture Notes in Control and Information Sciences*, 1991, vol. 600, pp. 226-251.

[9] **D.Makackas**. The use of the formal method for evaluation of performance characteristics of real-time systems, *Ph.D. Thesis*, 2004.

[10] **A.Andriulaitis**. Analysis of CORBA Load Balancing Strategies. *Information Technology and Control*, 2004, vol. 32, pp. 75-79.

INTRODUCING ABSTRACT DATA TYPES FOR MODELING THE STRUCTURAL CHANGES OF BIOLOGICAL SYSTEMS IN DYNPLA

Agne Paulauskaite-Taraseviciene[1], Henrikas Pranevicius[1], Kristina Sutiene[2]

[1]*Kaunas University of Technology, Department of Business Informatics, Studentu 56, Kaunas, Lithuania, agne@ifko.ktu.lt, henrikas.pranevicius@ktu.lt*
[2]*Kaunas University of Technology, Department of MathematicalResearch in Systems, Studentu 50, Kaunas, Lithuania, kristina.sutiene@ktu.lt*

Abstract. Addressing to various challenges in the field of genetics and life sciences, the evolutionary dynamic processes are modeled, as changes in the system's structure are the essential feature of such processes. Modeling of biological systems is particularly important, since it allows to investigate, predict and better understand the biological phenomena. We proceed with one of the formal method used for modeling and analysis of such systems, that of dynPLA formalism. In this paper, dynPLA method is extended by introducing abstract data types that allow for describing the structural changes of complex dynamic systems. The declared abstract data types are formalized using Object-Z notation. We illustrate this modeling approach by presenting both specification and verification of evolutionary, mutation and apoptosis processes observed in biological systems.

Keywords: PLA, variable structure, formalization, verification, biological systems.

1 Introduction

The ability of a system to dynamically change its structure play an important role in the studies of biological phenomena. The modeling of biological systems includes the description of processes, such as evolutionary, mutation, apoptosis, self-regulation and others [1]. The structure of such biological systems is described as complex network composition of various components (genes, proteins, neurons, cells, et al.). Moreover, in such systems the hierarchical configuration of its components is very important, since it follows the principles, such as organization, integration and others that are observed in biological systems [3]. Evolutionary processes of biological systems describe how this structure adapts dynamically in time to the internal and external factors [10].

Simulation of biological systems addresses various factors that may trigger the structural changes in the system's model and how the system behaves after changing its structure. Since such systems are complex and often crucial, the usage of formal methods is advantageous that enables the mathematical description and verification of a considered system. Among all possible formalisms, it is preferred to use techniques that are able to more accurately represent the structure of the system and describe its behavior in time [8, 9].

In this paper, the ability to use formal method dynPLA [4] is demonstrated by taking the example of certain biological system. dynPLA notation is extended by introducing abstract data types (ADT) that allow to facilitate the description of structural changes in the considered system.

2 Abstract data types in dynPLA

dynPLA is the extension of Piece Linear Aggregate (PLA) method [5] including dynamic features which allow to specify the structural changes in the system. In dynPLA, both aggregate and system of aggregates have a possibility to change their structure dynamically in time [4].

Definition. dynPLA is a structure $A_{dyn} = \langle A(t), R(t) \rangle$ with following constraints :

$\{A_1(t),...,A_n(t)\} \in A(t)$ – the set of aggregates ;

$R(t) = \{A_1(t),..., A_n(t)\} \times \{Y_1(t),..., Y_n(t)\} \rightarrow \{A_1(t),...,A_n(t)\} \times \{X_1(t),...X_n(t)\}$ – the coupling set ;

$\forall A_i(t) \in A(t) \mid A_i(t) = \langle X(t), Y(t), E(t), Z(t), H(t), G(t), A(t), R(t) \rangle$ – the structure of i^{th} aggregate ;

where $X(t)$ – the set of input signals ;

$Y(t)$ – the set of output signals ;

$E(t) = E'(t) \cup E''(t)$ – the set of events ;

$E'(t)$ – the set of external events ; $E''(t) = E_1''(t) \cup E_2''(t)$ – the set of internal events ;

$E_1''(t)$ – the set of state events; $E_2''(t)$ – the set of structure-change events ;

$Z(t) = \langle X(t), Y(t), E(t), \nu(t), z_\nu(t), H(t), G(t), A(t), R(t) \rangle$ – the state of aggregate ;

$z_\nu(t)$ – the continuous component ; $\nu(t)$ – the discrete component ;

$H(t)$ – the set of transition operators ;

$G(t)$ – the set of output operators ;

$A(t)$ – the set of internal aggregates ;

$R(t)$ – the coupling set among internal aggregates.

dynPLA model supports five types of structural changes in the aggregate system: addition of a new connection, removal of an existing connection, addition of a new aggregate, removal of an existing aggregate, and aggregate's migration from one place to another. The listed operations are general and independent of a given system's model. This is a motivation for introducing abstract data types (ADT)[2] to describe the structural changes in dynPLA specification. It allows to avoid the descriptions of many additional actions used in fulfillment any structural change and thereby present the specification in shorthand way; to produce an unambiguous statement for these actions needed while performing the structural changes and thus avoid specification errors, since one can manipulate only with operations that are defined in advance.

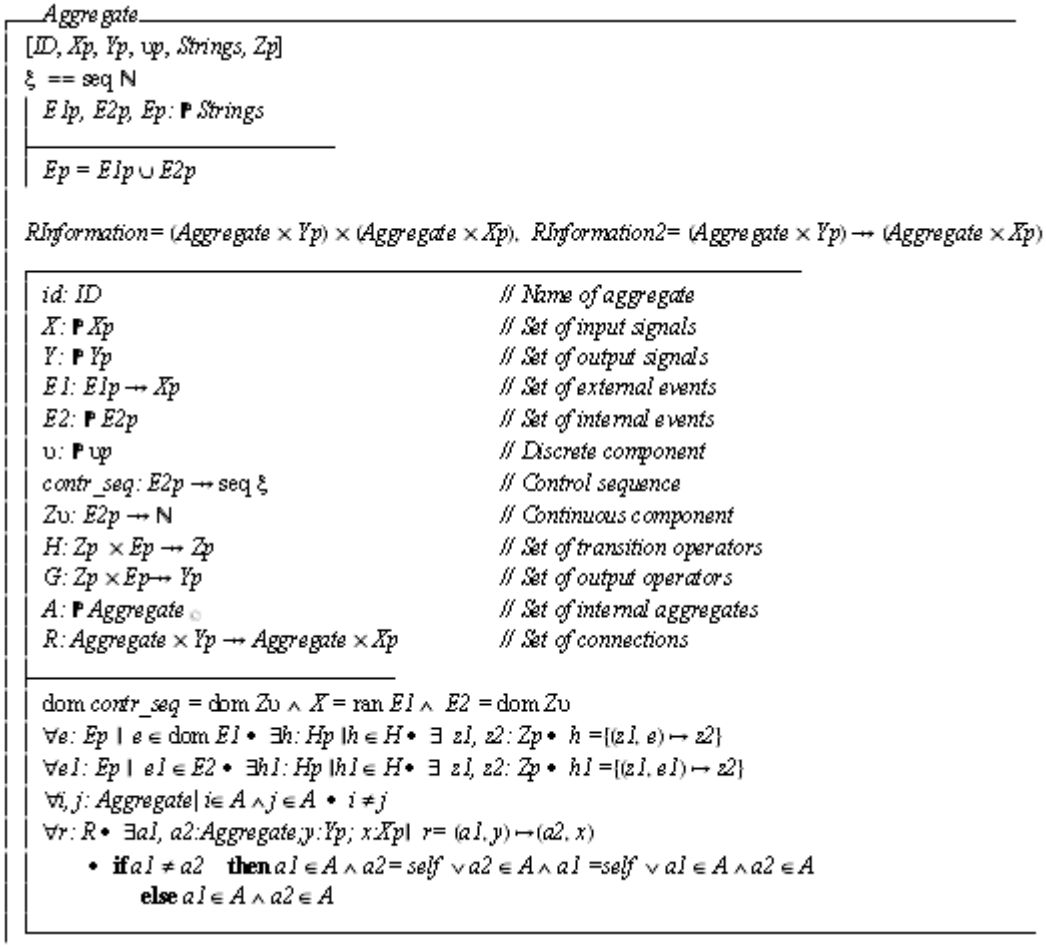


Figure 1. The state schema of class *Aggregate*

To define ADT in dynPLA, Object-Z specification language has been used [6]. This allows to unambiguously define the actions within operations, as well as to introduce the preconditions to be met in order to properly perform the structural changes in dynPLA model. Object-Z class *Aggregate* defines the dynPLA model structure (Figure 1) and its operations (Figure 2) for structural changes:

- *AddR* – add-link operation that combines three operations *Add_source, Add_target, Add_R* used to define the coupling *link?* in the source aggregate *a1* (addition of output signal *y* and output operator *G*), in the target aggregate *a2* (addition of input signal *x* and associated external event *e*, as well as operator *H* to process the input information), and in the external aggregate surrounding both of them (addition of coupling *link?*) ;
- *RemoveR* – remove-link operation that combines three operations *Remove_source, Remove_target, Remove_R* used to define the connection *link?* in the source aggregate *a1* (removal of output signal *y* and the output operator *G*), in the target aggregate *a2* (removal of input signal *x* and associated external event *e*, as well as operator *H* to process the input information), and in the external aggregate surrounding both of them (removal of coupling *link?*).

To add a new aggregate, three separate operations may be defined:

- *AddAg* – addition operation of a new aggregate which has no couplings with other aggregates at the moment of it's creation ;

- *AddAgR* – addition operation of aggregate and its connections. This operation can be described by combining operations used to add a new aggregate, as well as to couple new links. Dynamic changes of aggregate's structure are not significant at the beginning of its existence, since all signals and operators associated with newly created couplings may be included in aggregate's description in advance. For this purpose, it is better to define operations *AddAll_R, Expected_changes* that have no influence on the structure of newly attached aggregate while adding new links ;
- *ClounAg* – cloning operation of aggregate. During this operation the referred aggregate *a?* is cloned as a copy *cloun*, which has all the same settings as aggregate *a?* except its name. Operations *AddAll_R* and *Expected_changes* are used to create couplings for aggregate *cloun* that are analogous as aggregate's *a?* couplings.

$$
\begin{array}{l}
Add\_source \triangleq [\ \Delta(Y, G)\ link?: RInformation \mid \exists a1, a2: Aggregate;\ x: Xp;\ y: Yp;\ z: Zp;\ g: Gp;\ e: Ep \\
\quad \mid link? = (a1, y) \mapsto (a2, x) \wedge\ g = [(z, e) \mapsto y] \ \wedge y \in Y \wedge g \in G \\
\quad\quad \bullet\ Y' = Y \cup [y] \wedge G' = G \cup [g]\ ] \\
Add\_target \triangleq [\ \Delta(X, E1, H)\ link?: RInformation \mid \exists a1, a2: Aggregate;\ x: Xp;\ y: Yp;\ e: E1p; h: (Zp \times Ep) \times Zp; z1, z2: Zp \\
\quad \mid link? = (a1, y) \mapsto (a2, x) \ \wedge h = [(z1, e) \mapsto z2] \ \wedge\ x \in X \wedge e \mapsto x \in E1 \wedge\ h \in H \\
\quad\quad \bullet\ X' = X \cup [x] \wedge H' = H \cup [h] \wedge E1' = E1 \cup [e \mapsto x]\ ] \\
Remove\_source \triangleq [\ \Delta(Y, G)\ link?: RInformation \mid \exists a1, a2: Aggregate;\ x: Xp;\ y: Yp;\ z: Zp;\ g: Gp;\ e: Ep \\
\quad \mid link? = (a1, y) \mapsto (a2, x) \ \wedge g = [(z, e) \mapsto y] \wedge y \in Y \wedge g \in G \\
\quad\quad \bullet\ Y' = Y \mid [y] \wedge G' = G \mid [g]\ ] \\
Remove\_target \triangleq [\ \Delta(X, E1, H)\ link?: RInformation \mid \exists a1, a2: Aggregate;\ x: Xp;\ y: Yp;\ e: E1p;\ h: Hp;\ z1, z2: Zp \\
\quad \mid link? = (a1, y) \mapsto (a2, x) \ \wedge h = [(z1, e) \mapsto z2] \ \wedge\ x \in X \wedge e \mapsto x \in E1 \wedge h \in H \\
\quad\quad \bullet\ X' = X \mid [x] \wedge H' = H \mid [h] \wedge E1' = E1 \mid [e \mapsto x]\ ] \\
AddR \triangleq [\ \Delta(R)\ link?: RInformation \mid \exists\ a1, a2: Aggregate;\ x: Xp;\ y: Yp \\
\quad \mid link? = (a1, y) \mapsto (a2, x) \ \wedge\ (a1 \in A \wedge a2 = self \ \vee\ a2 \in A \wedge a1 = self \ \vee a1 \in A \wedge a2 \in A \wedge link? \in R) \\
\quad\quad \bullet\ R' = R \cup [link?] \ \wedge a1.\ Add\_source \ \wedge\ \ a2.\ Add\_target\ ] \\
AddAg \triangleq [\ \Delta(A)\ a?: Aggregate \mid a? \neq self \wedge\ a? \notin A \wedge\ A' = A \cup [a?]\ ] \\
AddAll\_R \triangleq [\ \Delta(R)\ a?: Aggregate,\ links?: RInformation2 \mid \forall r: links? \bullet (\exists a1, a2: Aggregate;\ x: Xp;\ y: Yp \\
\quad \mid r = (a1, y) \mapsto (a2, x) \wedge\ (a? = a1 \vee a? = a2) \ \vee (a? = a2 \wedge a? = a1)) \ \wedge\ a1 \in A \wedge a2 \in A \wedge r \in R] \\
\quad\quad \bullet\ R' = R \cup links?\ ] \\
Ag\_Remove\_no\_link \triangleq [\ \Delta(A)\ a?: Aggregate \mid (\forall r: R \bullet (\exists a1, a2: Aggregate;\ x: Xp;\ y: Yp \\
\quad \mid r = (a1, y) \mapsto (a2, x) \bullet a1 \neq a? \ \wedge a2 \neq a? \wedge a? \in A \wedge a? \neq self) \Rightarrow A' = A \mid [a?]\ ] \\
Remove\_All\_R \triangleq [\ \Delta(R)\ links?: RInformation2 \mid links? \subseteq R \wedge \forall r: links? \\
\quad\quad \bullet (\exists a1, a2: Aggregate;\ x: Xp;\ y: Yp \mid\ r = (a1, y) \mapsto (a2, x) \\
\quad\quad\quad \bullet a1.Remove\_source \wedge a2.Remove\_target \wedge\ R' = R \mid links?\ ] \\
RemoveR \triangleq [\ \Delta(R)\ link?: RInformation \mid \exists a1, a2: Aggregate;\ x: Xp;\ y: Yp \mid link? = (a1, y) \mapsto (a2, x) \\
\quad \wedge link? \in R \ \bullet\ R' = R \mid [link?] \wedge a1.Remove\_source \ \wedge a2.Remove\_target\ ] \\
Expected\_changes \triangleq [\ a?: Aggregate,\ links?: RInformation2 \mid \forall r: links? \bullet (\exists a1, a2: Aggregate;\ x: Xp;\ y: Yp \mid\ r = (a1, y) \mapsto (a2, x) \\
\quad\quad \bullet (\textbf{if}\ a1 \neq a? \wedge a2 = a?\ \ \textbf{then}\ a1.\ Add\_source \\
\quad\quad\quad \textbf{else}\ (\textbf{if}\ a2 \neq a? \wedge a1 = a?\ \ \textbf{then}\ a1.Add\_target))\ ] \\
Cloun \triangleq [\Delta(A)\ a?: Aggregate,\ links!: RInformation2,\ a!: Aggregate,\ id?: ID \\
\quad \mid a? \in A \wedge\ \exists cloun: Aggregate \mid cloun.X' = a?.X' \wedge\ \ldots \wedge cloun.id \neq a?.id \\
\quad\quad \wedge a?.id = id? \wedge cloun \notin A \wedge A' = A \cup [cloun] \\
\quad\quad \bullet \exists links: Aggregate \times Yp \rightarrow Aggregate \times Xp; \\
\quad\quad\quad s: Aggregate \times Yp \rightarrow Aggregate \times Xp \mid\ links \subseteq R \ \wedge s = \| \\
\quad\quad\quad \bullet (\forall r: links \\
\quad\quad\quad \bullet (\exists a1, a2: Aggregate;\ x: Xp;\ y: Yp \\
\quad\quad\quad\quad \mid r = (a1, y) \mapsto (a2, x) \wedge a1 = a? \vee a2 = a? \vee a1 = a? \wedge a2 = a? \\
\quad\quad\quad\quad \bullet (\textbf{if}\ a1 = a? \wedge a2 \neq a?\ \ \textbf{then}\ s = s \cup [((cloun, y) \mapsto (a2, x))] \\
\quad\quad\quad\quad\quad \textbf{else}\ (\textbf{if}\ a2 = a? \wedge a1 \neq a?\ \textbf{then}\ s = s \cup [((a1, y) \mapsto (cloun, x))] \\
\quad\quad\quad\quad\quad\quad \textbf{else}\ (\textbf{if}\ a1 = a? \wedge a2 = a?\ \ \textbf{then}\ s = s \cup [((cloun, y) \mapsto (cloun, x))] \\
\quad\quad\quad\quad\quad\quad\quad \textbf{else}\ s = s))))) \wedge links! = s \ \wedge a! = cloun\ ] \\
\\
Ag\_Remove\_pick\_links \triangleq [a?: Aggregate,\ links!: RInformation2 \ \mid a? \in A \wedge \forall rr: R \bullet (\exists a2, a3: Aggregate;\ x: Xp;\ y: Yp \\
\quad\quad \mid rr = (a2, y) \mapsto (a3, x) \wedge\ (a2 = a? \vee a3 = a? \vee a2 = a? \wedge a3 = a?) \ \bullet\ rr \in links!\ ] \\
AddAgR \triangleq AddAg \wedge AddAll\_R \wedge Expected\_changes\ ) \\
RemoveAg \triangleq Ag\_Remove\_no\_link\ [\!]\ ((Ag\_Remove\_pick\_links \parallel Remove\_All\_R)\ Ag\_Remove\_no\_link\ ) \\
ClounAg \triangleq Cloun \parallel (AddAll\_R \wedge\ Expected\_changes\ )
\end{array}
$$

Figure 2. Operations of class *Aggregate*

To remove an existing aggregate, the operation *RemoveAg* is applied. Removing the referred aggregate may cause two cases depending on whether it is merged with other aggregates or not. Operation *Ag_Remove_no_link* is used to remove the aggregate, which has no links with other aggregates. If aggregate *a?*

is connected with other aggregates, it must be removed not only the aggregate but also all its links. Operation *Ag_Remove_pick_links* gathers all links tied with eliminated aggregate. To remove these links, operation *Remove_All_R* is used.

All system's aggregates are denoted by class *All_Aggregates* (Figure 3). In this class, all possible structural changes with ADT *Aggregate* are also declared using the input information referred to the given operations.

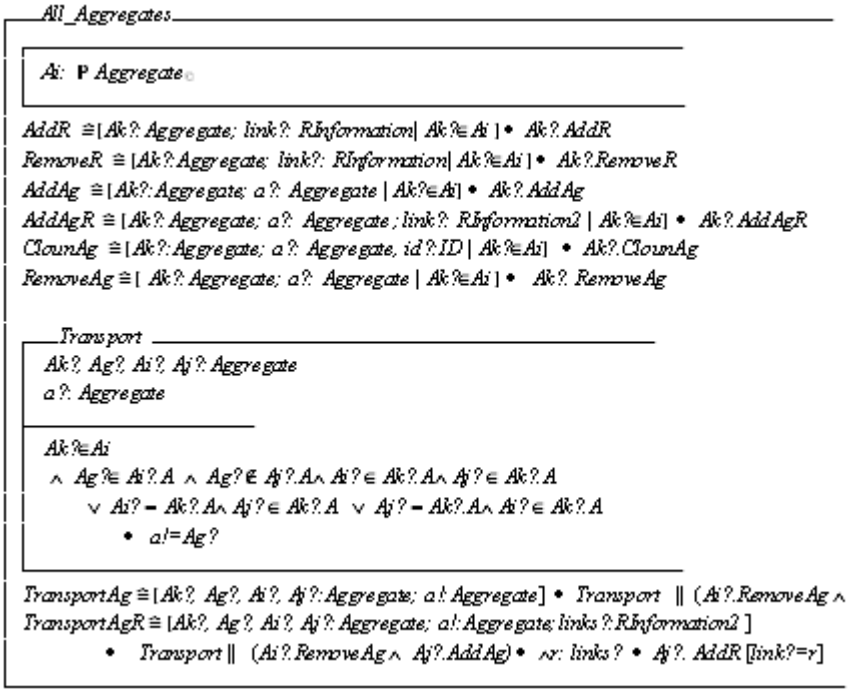


Figure 3. Object-Z class *All_Aggregates*

Aggregate's migration from one place to another is described by employing two operations already introduced above: aggregate's removal from aggregate system to which it belongs, and its addition to a new aggregate system. Aggregate's relocation may cause two scenarios that in turn are handled by two operations:

- *TransportAg* describes aggregate's migration from one system to another without adding any links to the new aggregate system ;
- *TransportAgR* describes aggregate's migration from one system to another by adding links to its new location.

3 Example: modeling the formation of cancer tumor and its spread

dynPLA model of a chosen biological system is demonstrated as the instance, where the formation of cancer tumor is described starting from the occurrence of cancer cells till the generation of metastasis [7]. This model is presented in an abstract level. To demonstrate the usage of ADT operations in dynPLA, the dynamic processes, such as cellular reproduction, apoptosis, and mutation are revealed.

3.1 Conceptual model

Cancer cells evolve from normal cells (Figure 4). This phenomenon is caused by genetic mutation that disrupts the cell development process, so that the growth of abnormal cells becomes uncontrolled (Figure 4).

Normal cells continue dividing if they get the internal signals with such information. This process is interrupted if internal functions are disturbed or the number of cells in tissue reaches a certain limit. Cancer cells ignore any signal that may stop the growth of normal cells, leading to the division of abnormal cells at faster rates. Almost all cells have a programmed cell death – apoptosis . During cell mutation, the response path *p53* is disabled, which controls the cell death processes. Disruption of this path results the loss of a cell ability to undergo apoptosis, i.e. cells become immortal. An uncontrolled division and proliferation of cancer cells eventually forms a mass known as a tumor (Figure 4).

When tumor reaches the certain its size, the cells located inside are under lower oxygen conditions. The tumor can activate the occurrence of new blood vessels that allows to achieve the nutrients. The tumor with blood vessels forms the invasive cancer (Figure 5).

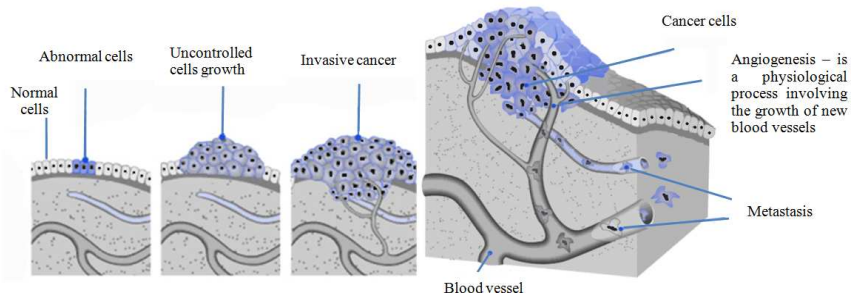


Figure 4. Cancer formation **Figure 5. Cancer metastasis**

In later stages, cancer cells can break away, leak, or spill from a primary tumor, enter lymphatic and blood vessels, circulate through the bloodstream, and be deposited within normal tissue elsewhere in the body, finally leading to the metastasis. Then, cancer cells begin colonize a new tissue and form therein a new tumor mass.

3.2 dynPLA model

In dynPLA model, each tissue is denoted by separate a aggregate AUD_i , which has links with other aggregates AUD_j, $i \neq j$, thus resulting the aggregate system *ORG* (Figure 6 a).

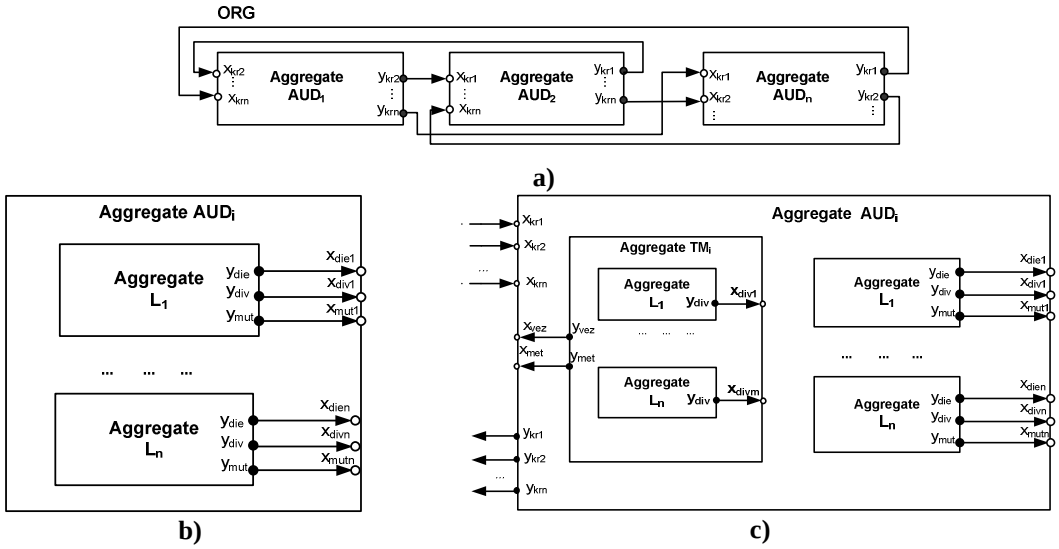


Figure 6. Connections among aggregates AUD_i a), structure AUD_i at initial time moment b) and with a tumor aggregate TM_i c)

Each aggregate AUD_i can locate one tumor aggregate TM_i and the set of aggregates L_j that denote separate cells. At the initial time moment, aggregate AUD_i contains aggregates L_j (Figure 6 b) . If cell mutation occurs, aggregate AUD_i locates inside the tumor aggregate TM_i, which consists of aggregates that describe the mutation cells (Figure 6 c).

1. $L_j = \langle X, Y, E, Z(t), H, G \rangle$
2. $X = \varnothing$ – input signals.
3. $Y = \{y_{die}, y_{div}, y_{mut}\}$ – output signals.
4. $E' = \varnothing$ – external signals.
5. $E_2'' = \{e_1'', e_2'', e_3''\}$ – internal signals.
6. // e_1'' – cell's division, e_2'' – cell's death, e_3'' – cell's mutation.
7. Controlling sequences: $e_1'' \mapsto \mu_k^1$ – duration of division
8. $e_2'' \mapsto \mu_1^2$ – duration of cell life-cycle.
9. $e_3'' \mapsto \mu_1^3$ – duration of cell mutation.
10. $z_v(t) = \{w(e_1'', t_m), w(e_2'', t_m), w(e_3'', t_m)\}$ – continuous component
11. State at initial time moment t_0:
12. $w(e_3'', t_0) = t_0 + \mu_1^3$, $w(e_1'', t_0) = t_0 + \mu_1^1$, $w(e_2'', t_0) = t_0 + \mu_1^2$.
13. $H(e_1'')$ // division of cell
14. $w(e_1'', t_{m+1}) = t_m + \mu_k^1$.
15. $G(e_1'') : Y = y_{div}$.
16. $H(e_2'')$ // cell's death
17. $w(e_1'', t_{m+1}) = \infty$, $w(e_2'', t_{m+1}) = \infty$
18. $w(e_3'', t_{m+1}) = \infty$.
19. $G(e_2'') : Y = y_{die}$.
20. $H(e_3'')$ // cell's mutation
21. $w(e_3'', t_{m+1}) = \infty$, $w(e_2'', t_{m+1}) = \infty$,
22. $w(e_1'', t_{m+1}) = t_m + (\mu_k^1 / h)$.
23. $G(e_3'') : Y = y_{mut}$.

Figure 7. Description of aggregate L_j

The specification of tissue aggregate AUD_i is presented in Figure 8. Aggregate AUD_i describes the structural changes in tissue that are caused by cell evolution and tumor formation. Input signal x_{divj} carries information that cell j is already divided and determines the duplication of cell aggregate L_j by employing operation *ClounAg* (20). The received signal x_{diej} informs that the referred cell j is dead. The elimination of aggregate L_j is described using operation *RemoveAg* (22). Signal x_{mutj} shows that cell j is after mutation (4). The description of mutation L_j is given based on the status of aggregate AUD_i. If this cell is the first one that undergoes mutation in aggregate AUD_i, the tumor aggregate TM_{ind} (27) with its couplings (28) is created using operation *AddAgR*. Aggregate L_j is transferred to the created aggregate TM_{ind} (29) with the aim of operation *TransportAgR*. The connection (30) between them is built, which is used to inform TM_{ind} about the division of L_j. If mutation occurs in aggregate L_j and aggregate TM_{ind} exists, then aggregate L_j is removed from AUD_i (32) using operation *RemoveAg*, sending it to the aggregate TM_{ind} by external signal (33-34).

$AUD_i(t) = \langle X(t), Y, E(t), Z(t), H(t), G(t), A(t), R(t) \rangle$,

1. $X(t) = \{x_{die_j}, x_{div_j}, x_{mut_j}, x_{vez}, x_{kr_j}, x_{met}\}$ – input signals,
2. $Y(t) = \{y_{kr_j}, y_{last}\}$ – output signals,
3. $E'(t) = \{e'_{die_j}, e'_{div_j}, e'_{mut_j}, e'_{vez}, e'_{kr_j}, e'_{met}\}$ – external signals.
4. $E'' = \varnothing$ – internal signals.
5. $v(t) = \{SV(t), GL(t), ST(t)\}$ – discrete components,
6. $SV(t)$ – concentration (number) of normal cells,
7. $ST(t) = \begin{cases} 0, & not\ cancer, \\ 1, & cancer. \end{cases}$ $GL(t) = \begin{cases} 0, & not\ cancer, \\ 1, & cancer. \end{cases}$
8. $A(t) = \{L, TM_{ind}\}$ – internal aggregates,
9. // TM_{ind} – tumour aggregate,
10. // $L(t) \subseteq \{L_1, L_2, .., L_n\}$ – set of cells, $SV(t) = {}^{\#}L(t)$.
11. $R(t) = \begin{Bmatrix} L_j, y_{die} \to AUD_i, x_{die_j} \\ L_j, y_{div} \to AUD_i, x_{div_j} \\ L_j, y_{mut} \to AUD_i, x_{mut_j} \\ TM_{ind}, y_{vez} \to AUD_i, x_{vez} \\ TM_{ind}, y_{met} \to AUD_i, x_{met} \\ AUD_{ind}, y_{last} \to TM_{ind}, x_{last} \end{Bmatrix}$.
12. $H(t) = \{H(e''_1), H(e'_{die_j}), .., H(e'_{met})\}$ –transition operators.
13. $G(t) = \{G(e''_1), G(e'_{die_j}), .., G(e'_{met})\}$ –output operators.
14. The state at initial time moment t_0:
15. $GL(t_0) = 0$, $SV(t_0) = k$, $ST(t_0) = 0$, $Y(t_0) = y_{kr_j}$, $j = \overline{1,k}$,
16. $A(t_0) = \{L(t_0)\}$, ${}^{\#}L(t_0) = k$, $E'(t_0) = \{e'_{div_j}, e'_{die_j}, e'_{mut_j}\}$,…
17. $R(t_0) = \begin{Bmatrix} L_j, y_{die} \to AUD_i, x_{die_j} \\ L_j, y_{div} \to AUD_i, x_{div_j} \\ L_j, y_{mut} \to AUD_i, x_{mut_j} \end{Bmatrix}$.
18. $H(e'_{div_j})$:// *Received signal about division of cell j*
19. *if* ${}^{\#}SV(t_m) < s \wedge ST(t_m) = 0$ *then*
20. $AUD_i(t_{m+1}) = ClounAg(AUD_i(t_m), L_j, L_{new})$,
21. $H(e'_{die_j})$: // *Received signal about death of cell j*
22. $AUD_i(t_{m+1}) = RemoveAg(AUD_i(t_m), L_j)$,
23. $G(e'_{die_j}) : Y = \varnothing$.
24. $H(e'_{mut_j})$: // *Received signal about mutation of cell j*
25. *if* $GL(t_m) = 0 \wedge ST(t_m) = 0$ *then*
26. $GL(t_{m+1}) = 1$,
27. $AUD_i(t_{m+1}) = AddAgR(AUD_i(t_m), TM_{ind}, r)$,
28. *where* $r = \begin{Bmatrix} TM_{ind}, y_{vez} \to AUD_i, x_{vez} \\ AUD_i, y_{last} \to TM_{ind}, x_{last} \end{Bmatrix}$,
29. $AUD_i(t_{m+1}) = TarnsportAgR(AUD_i(t_m), L_j, AUD_i, TM_{ind}, r)$
30. *where* $r = \{L_j, y_{div} \to TM, x_{div_j}\}$,
31. *else*
32. $AUD_i(t_{m+1}) = RemoveAg(AUD_i(t_m), L_j)$,
33. $G(e'_{mut_j}) : y_{last} = VEZ$, *if* $GL(t_m) = 1 \wedge ST(t_m) = 0$,
34. *where* $VEZ = L_j$.
35. $H(e'_{vez})$: // *Received signal about cancer spread*
36. $ST(t_{m+1}) = 1$,
37. $AUD_i(t_{m+1}) = AddR(AUD_i(t_m), r)$,
38. *where* $r = \{TM_{ind}, y_{met} \to AUD_i, x_{met}\}$,
39. $G(t_{m+1}) = G(t_m) \cup G(e'_{met})$,
40. $H(e'_{met})$: // *Metastasis in other tissues*
41. $G(e'_{met}) : y_{kr_j} = VEZ$, $j = rand(1, n)$.
42. $H(e'_{kr_j})$: // *Metastasis occurred in the tissue*
43. *if* $GL(t_m) = 0 \wedge ST(t_m) = 0$ *then*
44. $GL(t_{m+1}) = 1$,
45. $AUD_i(t_{m+1}) = AddAg(AUD_i(t_m), VEZ)$,
46. $AUD_i(t_{m+1}) = AddAgR(AUD_i(t_m), TM_{ind}, r)$,
47. *where* $r = \begin{Bmatrix} TM_{ind}, y_{vez} \to AUD_i, x_{vez} \\ AUD_i, y_{met} \to TM_{ind}, x_{met} \end{Bmatrix}$,
48. $AUD_i(t_{m+1}) = TarnsportAgR(AUD_i(t_m), VEZ, AUD_i, TM_{ind}, r)$
49. *where* $r = \{L_j, y_{div} \to TM_{ind}, x_{div_j}\}$, $L_j = VEZ$,
50. $G(e'_{kr_j}) : y_{last} = (VEZ)$, *kai* $GL(t_m) = 1 \wedge ST(t_m) = 0$.

Figure 8. **Description of aggregate AUD_i**

Input signal x_{vez} informs that the tumor evolves as cancer (35-39), resulting the couplings for possible metastasis from TM_{ind} (37-38). Using signal x_{met} (40-41), aggregate AUD_i gets the aggregates (VEZ) detached from TM_{ind}. These aggregates (VEZ) are forwarded to randomly chosen aggregate AUD_j. The aggregate AUD_i can also get the aggregates (VEZ) of cancer cells from the other aggregates that denote tissue (42-50).

Aggregate TM_i describes the behaviour of tumor (Figure 9). This aggregate consists of a set V of aggregates for mutated cells (13). Since each of these aggregates has only internal event for division, the aggregate TM_i creates a copy of aggregate using operation *ClounAg* (21) if TM_i gets the signal x_{divj}. When the size $D(t)$ of tumor (10) reaches the constant k (21), the amount of cancer cells converges, leading to the situation when the calculation of them becomes meaningless and an internal structural event is initiated – formation of invasive cancer (23). In the event of cancer (29-32), the continuous component is activated for generation of

metastases (31), at the same time one of internal aggregates (37) is removed and sent to it surrounding aggregate AUD_i using signal (38).

$TM_i(t) = \langle X(t), Y(t), E(t), Z(t), H(t), G(t), A(t), R(t) \rangle$,

1. $X(t) = \{x_{div_j}, x_{metin}\}$ – input signals.
2. $Y(t) = \{y_{vez}, y_{met}\}$ – output signals.
3. $E'(t) = \{e'_{div_j}\}$ – external events.
4. $E''_2 = \{e''_1, e''_2\}$ – internal events for structural changes ,
5. // e''_1 – cancer formation , e''_2 – metastasis formation.
6. Controlling sequences:
7. // $e''_1 \mapsto \mu_1^1$ – duration of cancer formation,
8. // $e''_2 \mapsto \mu_k^2$ – duration of metastasis formation,
9. $v(t) = D(t)$ –discrete component ,
10. //$D(t)$ –size of tumour (amount of cells), $D(t) = {}^{\#}V(t)$.
11. $z_v(t) = \{w(e''_1, t_m), w(e''_1, t_m)\}$ – continuous component.
12. // $A(t) = V(t)$ – internal aggregates .
13. // $V(t) \subseteq \{L_1, L_2, .., L_n\}$ – set of cells.
14. $R(t) = \{L_j, y_{div} \to TM_i, x_{div_j}\}$ – internal couplings.
15. $H(t) = \{H(e''_1), H(e''_2), H(e'_{div_j}), H(e'_{last})\}$ –transition operators.
16. $G(t) = \{G(e''_1), G(e''_2), G(e'_{div_j}), G(e'_{last})\}$ –output operators.
17. State at initial time moment t_0: $z_v(t_0) = \{\infty, \infty\}$, $X(t_0) = \varnothing$
18. $E'(t_0) = \varnothing$, $Y(t_0) = \{y_{vez}\}$, $H(t_0) = \{H(e''_1), H(e''_2)\}$,
19. $G(t_0) = \{G(e''_1)\}$, $R(t_0) = \varnothing$, $A(t_0) = \varnothing$.
20. $H(e'_{div_j})$: // *Division of cancer cells*
21. $TM_i(t_{m+1}) = ClounAg(TM_i(t_m), L_j, L_{new})$, *if* $D(t_m) < k$,
22. *if* $D(t_m) \geq k \wedge w(e''_1, t_m) = \infty \wedge w(e''_2, t_m) = \infty$ *then*
23. $w(e''_1, t_{m+1}) = t_m + \mu_1^1$,
24. *else*
25. $w(e''_1, t_{m+1}) = w(e''_1, t_m)$,
26. $H(e'_{last})$: // *Receipt of cancer cell during metastasis*
27. $TM_i(t_{m+1}) = AddAgR(TM_i(t_m), VEZ, r)$, *if* $D(t_m) < k$,
28. *where* $r = \{L_j, y_{div} \to TM_i, x_{div_j}\}$, $L_j = VEZ$.
29. $H(e''_1)$: // *Cancer formation*
30. $w(e''_1, t_{m+1}) = \infty$,
31. $w(e''_2, t_{m+1}) = t_m + \mu_k^2$.
32. $G(e''_1): Y = y_{vez}$.
33. $H(e''_2)$: // *Metastasis formation*
34. $w(e''_2, t_{m+1}) = t_m + \mu_k^2$,
35. $TM_i(t_{m+1}) = ClounAg(TM_i(t_m), L_j, L_{new})$,
36. where $L_j = rand(V(t_m))$
37. $TM_i(t_{m+1}) = RemoveAg(TM_i(t_m), L_j)$.
38. $G(e''_2): y_{met} = (VEZ)$, $VEZ = L_j$.

Figure 9. Description of aggregate TM_i

3.3 Model verification based on graph of accessible states

The dynPLA model presented in Section 3.2 is checked using the verification approach – graph of accessible states [5].

The graph vertices of aggregate L_i are composed from three coordinates: 1. Cell division $w(e''_1)$; 2. Cell apoptosis $w(e''_2)$; 3. Cell mutation $w(e''_3)$. Since aggregate's L_i structure is fixed, graph vertices are made of the same components.

The graph vertices of aggregate TM_i are composed from the following coordinates: 1. The number of cancer cells *D(t)*; 2. The formation of cancer $w(e''_1)$; 3. The appearance of metastases $w(e''_2)$; 4. The internal aggregate's L_j, $j = \overline{1,n}$ state: $(1,0,0)^n$, where $(1,0,0)^n$ – coordinates of internal aggregates, $(1,0,0)^n = \underbrace{(1,0,0),(1,0,0),..,(1,0,0)}_{n}$, *n=D(t).*

The graph vertices of tissue aggregate AUD_i are composed from such coordinates: 1. The number of normal cells *SV(t)*; 2. The state of tissue *GL(t)*; 3. The state of tumor $ST(t)$; 4. The state of each aggregate L_j, $j = \overline{1,n}$; 5. The state of aggregate TM_i. The structural changes in aggregate AUD_i determine that the set of coordinates for graph vertices can vary (Figure 10.).

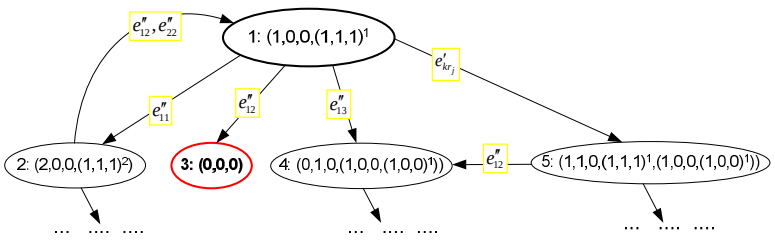
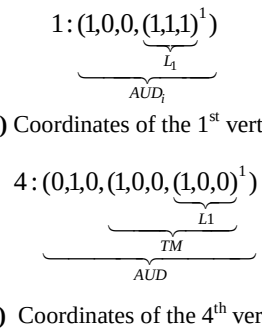


Figure 10. The fragment of state graph for aggregate AUD_i



a) Coordinates of the 1st vertex

b) Coordinates of the 4th vertex

Figure 11. Coordinates of graph vertices

The first vertex of given graph describes the initial state of aggregate if ${}^{\#}L(t_0)=1$, determining the coordinates as shown in Figure 11 a. If the internal event e''_{13} (mutation) for a cell L_1 occurs, the external signal is sent, resulting the migration to a new state –the fourth one based on operators $H(e''_{13})$ and $H(e'_{div_1})$. The actions performed as in operator $H(e''_{13})$ change the aggregate's state to a value $(1,0,0)$. The fulfillment of operator $H(e'_{div_1})$ determines that aggregate TM_i is created using operation *AddAgR*, as well as its coordinates (0,0,0) are added. Employing the operation *TransportAgR*, aggregate L_1 is moved to aggregate TM_i, thus the state extends to a value $(1,0,0,(1,0,0)^1)$. Finally, the coordinates of the fourth state obtain a value as given in Figure 11 b. If the internal event e''_{11} (division) for a aggregate L_1 occurs, the new aggregate L_2 with coordinates (1,1,1) is created using operation *AddAg*. The new vertex $2:(2,0,0,(1,1,1)^2)$ is reached. During the internal event e''_{12} (apoptosis), aggregate L_1 is removed based on operation *RemoveAg*, as well as the new state $3:(0,0,0)$ is obtained. In the same way the other coordinates of graph vertices are varying, since they are under influence of operations for structural changes used in specification.

4 Conclusions

Based on the definition of extended dynPLA model, each dynPLA model can exist as separate aggregate at one time moment, and as aggregate system at other time moment. This allowed to naturally describe complex biological processes, such as the division of cells, cancer spread and others. The introduced ADT enabled to represent the number of actions used to perform structural changes as separate operations, and thus to shorten the description of dynamic changes in a system. In addition, there is a possibility to extend the specification by introducing the new operations if needed, that greatly increase the flexibility of structure-change modeling. The formal ADT description was presented that allowed to specify dynPLA structural changes as object-oriented and in unambiguous way.

The model of biological system, such as the cancer spread in tissues, was presented to demonstrate the usage of ADT to describe the structural changes, such as the formation of tumor (*AddAgR*), the death of cells (*RemoveAg*), the division of cells (*CloungAg*), the migration of cells (*TransportAgR*). The graph of accessible states was created to ensure the correctness of model specification with data abstractions used to describe the structural changes in the modeled system.

References

[1] **Cassman M., et.al**. Report on International Research and Development in Systems Biology. *World Technology Evaluation Center*. 2005.

[2] **Dale, N.; Walker, H.** Abstract Data Types: Specifications, Implementations and Applications. *Heath*, 1996, p. 752.

[3] **Maus C., Mathias J., Mathias R., Uhrmacher A.M.** Hierarchical Modeling for Computational Biology. *Lecture notes, University of Rostock*. 2008.

[4] **Packevičius S., Kazla A., Pranevičius H.** Extension of PLA Specification for Dynamic System Formalization. *Journal Information Technology and Control*. 2006, volume 35, No.3, p. 235-242.

[5] **Pranevicius H**. Formal specification and analysis of Computer Net Protocols: aggregate method. *Technologija: Kaunas*. 2003.

[6] **Smith G**. The Object-Z Specification Language. *Kluwer Academic Publishers*. 2000.

[7] **Strazdaite V**. Cancer cells more closely. *Biophysics seminar, University of Vilnius*. 2006. (*in Lithuanian*)

[8] **Uhrmacher A.M., Himmelspach J., Röhl M ., Ewald R.** Introducing Variable Ports and Multi-Couplings for Cell Biological Modeling in DEVS. *Proceedings of the 2006 Winter Simulation Conference.* 2006, p. 832-840.

[9] **Uhrmacher A. M.; et.al**. Combining Micro and Macro-Modeling in DEVS for Computational Biology. *Proceedings of the 39th conference on Winter simulation*. 2007, p. 871-880.

[10] **Wolkenhauer O**. System Biology: Dynamic Pathway Modeling. *Systems Biology & Bioinformatics Group, University of Rostock*. 2008.

HIGH SCHOOL TIME TABLE PROBLEM SOLVING AND COMPARISON WITH AUTOMATIC OPTIMIZED TIMETABLE

Lina Pupeikiene[1], Jonas Mockus[2]

[1]*Vilnius Gediminas Technical University, Sauletekio al. 11, LT-10223 Vilnius, Lithuania, Lina.Pupeikiene@gmail.com*

[2]*Institute of Mathematics and Informatics, Akademijos g. 4, LT-08663 Vilnius, Lithuania, JMockus@gmail.com*

Abstract. In every high school it is most important to build school schedule for high school pupils. Every pupil of high school can choose individual lessons and has individual schedule. The problem is more complicated when the every pupil has possibilities to choose not only subjects, but hour per week of this subject too. However, as the number of teachers, number of pupils, number of different subjects, number of different subject hours, time slots and the constraints increases, the required time to find at least one feasible solution grows exponentially. All pupils, which have the same subject, are grouped to the subject-groups. Every subject-group has own teacher. The high school schedule is created from these subject-groups. The article includes such aims: development and estimation of real high school timetables program; comparison results of real high school schedule and automatically optimized schedule.

Key words: High school timetabling, optimal scheduling, subject-groups forming, heuristics evaluation.

1 Introduction

A timetable specifies which people meet at which location and at what time. The timing of events must be such that nobody has more than one event at the same time. School timetabling as a term refers to the construction of weekly timetables for schools of secondary education [14]. Specific feature of school timetabling field is a great number of research papers and widely used commercial software. Therefore a discussion of new results will be.

The events are lessons in a subject, taught by a teacher to a group of pupils in a single room. The timetable assigns a teacher, a pupils group, a room, and a time slot to each lesson. The pupil groups are specific to the subject, we call them subject-groups. A high school is referred here as the last grades of a high school or gymnasium where the pupils can mostly choose their preferred learning profile subjects. Therefore, this task is more complex in comparison with a secondary school scheduling without high school classes.

Some combinations of assignments lead to acceptable timetables, constraints follow from conditions imposed by rooms, pupils or teachers. We distinguish two types of constraints: conditions that must be met ("hard" constraints) and desires that should be fulfilled as well as possible ("soft" constraints). An important set of soft constraints is defined by didactic reasons. For example, by placing "hard" subjects, such as mathematics or physics, into morning hours. The maximal number of daily hours Tmax is obviously a hard constraint. Timetabling can be generally defined as the activity of assigning, subject to constraints, a number of events to a limited number of time periods and locations such, that desirable objectives are satisfied as nearly as possible [26]. Educational timetabling can be divided into three main classes: school timetabling, course timetabling and exam timetabling [15]. The goal is to find a timetable that satisfies all the hard constraints and minimizes the violation of soft constraints.

2 Overview of publications

A survey on educational timetabling problems [23] gives an overview of the literature. Overviews on examination timetabling and university course timetabling are in [4, 12, 13]. A comprehensive overview of formulations and of state-of-the-art approaches is in the surveys [4, 7, 8, 13, 15], in the proceedings of the PATAT conferences [5 – 7, 9, 10] and in the Lecture Notes in Computer Science series [9 – 11]. The European working group on automated timetabling (EURO-WATT) maintains a website with information on timetabling problems [25].

3 New Elements

The first new element of this work is the application and systematic investigation of the Bayesian Heuristic Approach [20] for optimization of heuristic parameters. These include the initial temperature and the cooling rate of Simulating Annealing (SA) algorithm and the randomization parameter of the local search algorithm. The formulation of the objective function in terms of Pareto optimality seems to be new in the field of school scheduling. The paper describes apparently the first web-based platform-independent implementation of

the software. Java servlet provides conditions for application at any school with internet connection. Any web browser works, no additional software is needed. Note that efficiency of recent versions of Java is close to that of the most efficient programming languages [9].

4 Defining Optimization Problem

Ministry of Education of the Republic of Lithuania has confirmed basic rules for high school schedule forming. They can be complementary of each school's rules and restrictions. However, the main purpose of these limitations is to develop a schedule, which would evaluate of the Ministry of Education requirements. In addition, this schedule must be acceptable to both: pupils and teachers.

Required schedule restrictions (formed by the Ministry of Education):

1. Working days d per week must be $d \leq 5$.
2. The teacher simultaneously cannot work in several different places.
3. The teacher cannot have more than 36 hours per week.
4. The pupil simultaneously cannot learn few different subjects.
5. A pupil i may have $28 \leq i \leq 32$ lessons per week.
6. It cannot be more then $p \leq 7$ lessons p per day.
7. Number of pupils i in one subject-group can be $15 \leq i \leq 30$.
8. In each classroom simultaneously cannot be several different types of subjects (for example, mathematics and physics).
9. Subjects, requiring special measures or facilities, shall be taught in the special classrooms (for example, IT, chemistry etc.).

Technically any required restriction violations cannot be broken. There can be only some minor offenses necessary restrictions, if it significantly improves the quality of the schedule. To define with timetable is good or bad we use penalty points. The penalty point's c_r, which assessing these restrictions, should be imposed very strictly.

The main required penalty point's restrictions function is as follows:

$$F_f = \sum_r c_r N_r \,, \tag{1}$$

here c_r – penalty for required restriction r; N_r – number of required restriction. In this case $r = 1, .., 9$.

Some of required restrictions c_r can be evaluated by the individual rules of each school. Such requirements are called needful, or "soft" constrains. They are valued differently in each school.

The main needful restrictions of the schedule include:

- Elimination of "windows" in teacher's schedule.
- Elimination of "windows" in pupil's schedule.
- Unacceptable working hours.
- Unacceptable workdays.
- Unacceptable order of subjects.
- Changing of pupils in the formed subject-group.

Usually penalty points for these restrictions are as follows:

c_m – penalty for the "window" on teacher's m schedule.

c_s – penalty for the "window" on pupils s schedule.

c_{mv} – penalty for "bad" hour v of teacher m.

c_{md} – penalty for "bad" day d of teacher m.

c_{sv} – penalty for "bad" hour v of pupil s.

c_{pd} – penalty for violation of pedagogical didactic pd.

c_{mg} – penalty of the list change of subject-group g taught by teacher m.

"Bad" hour/day is the hour/day, when teacher/pupil already has a work hour. Pedagogical didactic evaluates the difficulty of subjects. Most difficult subjects must be in the 1-4 lessons during the day. Less important subjects – in the end of the day. The importance of every subject is written in initial data file.

The sum function of the needed restrictions penalty points is as follows:

$$F_n = \sum_m c_m L_m + \sum_s c_s L_s + \sum_m \sum_v c_{mv} L_m^v + \\ + \sum_m \sum_d c_{md} L_m^d + \sum_s \sum_v c_{sv} L_s^v + \sum_{pd} c_{pd} L_{pd} + \sum_n c_{mg} L_n, \quad (2)$$

here L_m – number of "windows" on teachers m schedule; L_s – number of "windows" on pupils s schedule; L_m^v – number of "bad" hours v on the teachers m schedule; L_m^d – number of "bad" days d on the teachers m schedule; L_s^v – number of "bad" hours v on the pupils s schedule; L_{pd} – number of pedagogical didactic pd violations; L_n – number n of changing formed subject-group.

All physical restrictions and inconveniences are showed in Figure 1.

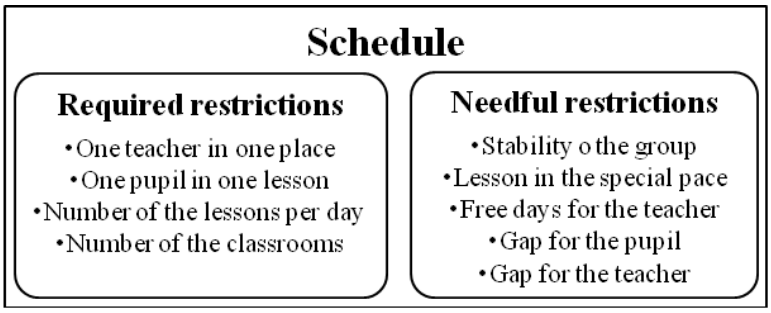


Figure 1. Restrictions for a creation of high school schedule

A compromise solution is reached by defining penalties for violation of constraints and disregarding inconveniences. Therefore, penalty points are calculated:

$$F = F_f + F_n. \quad (3)$$

where, F_f – is a sum of the penalties for the required restrictions; F_n – is a sum of the penalties for the needful restrictions (disregarding inconveniences). Optimal schedule will be schedule, which has as less as possible penalty points. To find such schedule, objective function F should be optimized. To not analyze the schedules with same number of penalty points, Pareto optimality was formulated. So we will get less variants to analyze and will save the users time. The optimization problem is

$$\min_{\tau \in A} F(\tau), \quad (4)$$

where, $F(\tau)$ is the total penalty of some schedule τ; A is the set of schedules satisfying the physical constraints. The penalties $F(\tau)$ depend on expert evaluations, therefore we regard them as heuristics.

5 "School schedule optimization" program working steps

"School schedule optimization" program designed to high school scheduling.

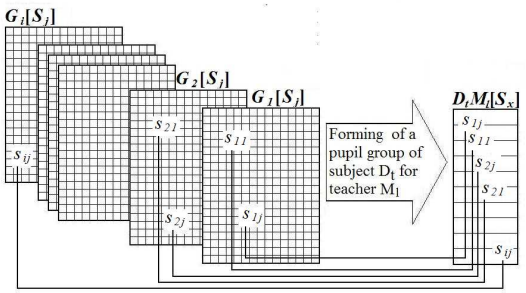


Figure 2. Forming subject-groups to teachers

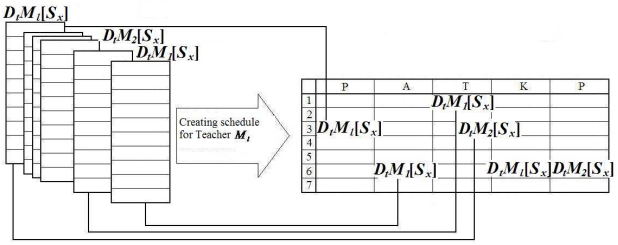


Figure 3. Time table for teachers creation

Figure 2 illustrates how subject-groups are assigned to teachers. Here pupils s_{ij} from groups G_i are grouped to the groups with identical subject D_t. Identical subject has same name and same hours per week. These groups are called subject-groups (with x pupils in the group) and assigned to the teacher M_l. Figure 3 shows how teacher's timetables are created. The subject-groups $D_tM_l[S_x]$, with teacher M_l, subject D_t and pupils of this subject-group S_x, are putted to the free class-room and to school timetable. When process is finalized, the optimization process is ready to start.

After optimizing, we can see such results of this program:

- school schedule;
- individual pupils schedules;
- individual teachers schedules;
- individual room schedules;
- subject-group schedules;

All results user can see in the program (on working time), or download them as archive personal computer. The program does not require much effort to the user, the payment to work with a computer, or a lot of time to understand how system works.

6 Optimization Methods

6.1. Defining Neighbourhood

Many different definitions can be used defining neighbourhood in a set A of feasible timetables *d*. The definition is important because local search is performed in the neighbourhood of the given point. We search for better timetables by subsequent closing of gaps for pupils and teachers. In this case the neighbours of a timetable *d'* are all timetables *d"* that can be reached from *d'* by a sequence of closing gap operations. This way we obtain locally optimal *d*(d')* that depends on the initial point *d'*.

Local search can be randomized by selecting current candidate (a pupil or a teacher) for gap closing with some probability x_0. Closing gaps for randomly selected pupils and teachers, we modify the search sequences. However, this not helps to reach the global optimum since the neighbourhood remains the same.

6.2. Escaping Neighbourhood

Simplest algorithm to search for global optimum is just random search with uniform distribution of observations (observation is calculation of the objective function at some fixed point). The advantages are simplicity and convergence to a global minimum of continuous functions. A well-known way to escape the local minimum is Simulated Annealing [1, 2, 14, 19, 21, 22]. Denote

$$\delta_n = F(d^{n+1}) - F(d^n), \tag{5}$$

Here d^n is a current timetable, d^{n+1} is a new timetable generated by closing gap operation. Define the probability

$$p_n = e^{\frac{-\delta_n}{x_1/\ln(1+x_2 n)}}, \ if \ \ \delta_n > 0, \tag{6}$$

$$p_n = 1, \ if \ \ \delta_n < 0, \tag{7}$$

where parameter x_1 is the "initial temperature", parameter x_2 defines the "cooling rate". SA algorithm means:

$$go\ to\ new\ timetable\ d^{n+1}\ with\ probability\ p_n \tag{8}$$

To apply the SA to a specific problem, one must specify the parameters x_1 and x_2.The choice can have a significant impact on the method's effectiveness. Unfortunately, there are no choices of these parameters that will be good for all problems.

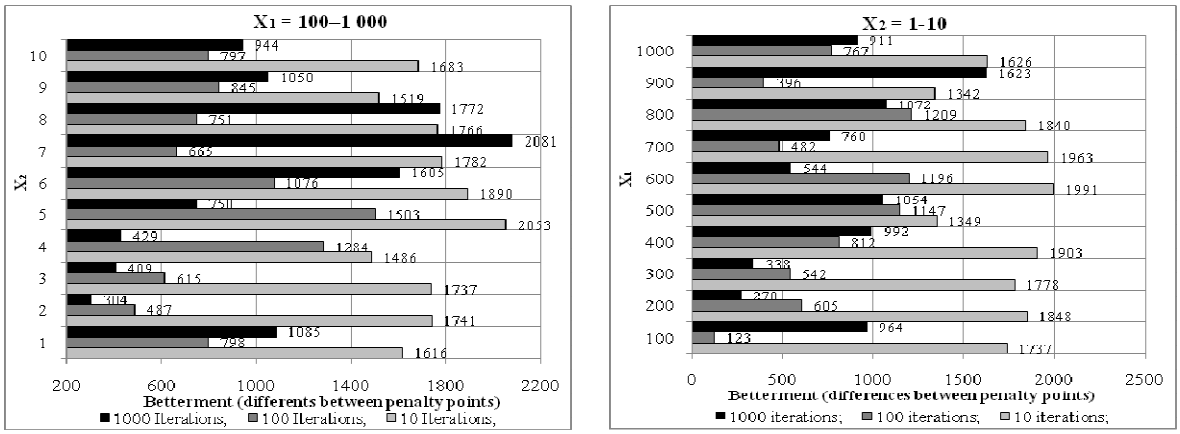


Figure 4. The best results of SA using different parameters

Analyzing Figure 4, we see different results using different initial parameters. Here difference of penalty points (between initial and optimal schedules) is calculated. Every column is received after *100* experiments with fixed initial parameters (Iterations, x_1 and x_2). In the left side of Figure 2 the results are grouped by x_2 when x_1 was between *100* and *1000*. In the right side, the results are grouped by x_1 when x_2 was between *1* and *10*. There are showed only best results after.

6.3. Bayesian Heuristic Approach

The Bayesian Heuristic Approach was designed for automatic optimization of heuristic parameters by filtering the noise during optimization of multi-modal functions [20]. We need to optimize three heuristic parameters $x = (x_0, x_1, x_2)$. Optimal parameters are obtained using the data of some specific school.

We cannot see optimal parameters x_1, x_2 of SA. Optimal results depend on the initial soft constrains and number of iteration. A way to adapt these parameters to a given problem is automatic optimization. This is not an easy problem since we need optimize multi-modal function with considerable noise. Here the Bayesian Heuristic Approach (BHA) [20] is useful.

Figures 5 and 6 illustrate efficiency of automatic adaptation of SA parameters using BHA. In these figures, the difference between initial and optimal timetable is showed. There we see *100* experiments with every different SA iteration. SA parameters were set automatically. Figure 3 shows, that method is more efficient as more SA iteration are used. Figure 4 illustrates the best results what was shown during *100* experiments with every different SA iteration. There we can see, that the best results we will get when it will be many SA and BHA iterations.

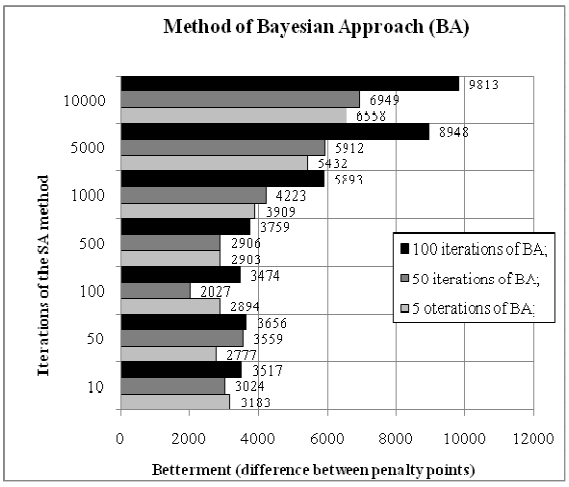


Figure 5. Average of 100 experiments results using BHA

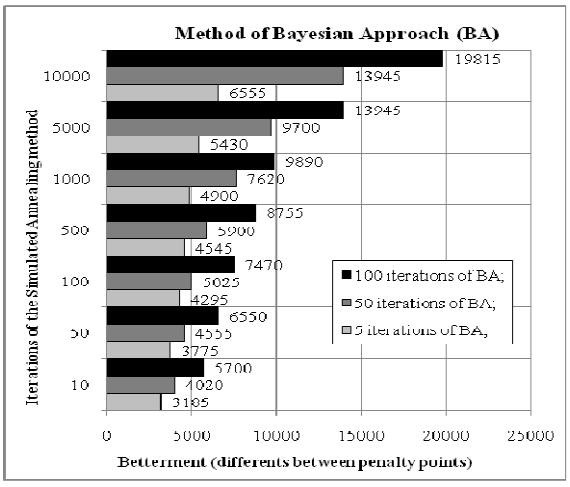


Figure 6. The best results of BHA after 100 experiments with each different SA iteration

However, the results can be used in similar schools as an approximation.

7 Comparison of results

Here are compared such results: real schedule created in a Lithuanian high school and, from pupils and teachers wishes, created and optimized schedule. Schedule was automatically optimized with Bayes method the results we can see in Figure 7. Both, schedule and data are from the same school and same classes.

Evaluating both types of schedules, penalty points were calculating for:

- pupil window – 5;
- teacher window – 300;
- teachers wished free time – 10;
- exceeding maximum hour limit – 2000;
- pedagogical didactic – 5.

Sum of seted penalty points for the real schedule was **380 020**. It is always same, because after finishing the creation process it can't be changed. Sum of penalty points after optimization process (was seted same penalty points) are showed in the Figure 5. There are few results after optimization with different initial parameters of optimisation method Bayes. The results are different while every time schedule is created from the new point.

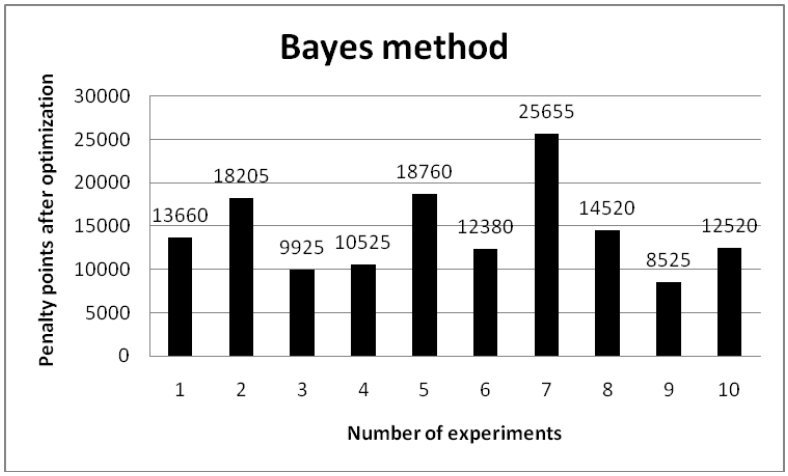


Figure 7. Penalty points after creating and optimizing schedule from initial data file

As we can see, the optimization results are much better as real schedule result. It is so, while optimization program creates and optimizes schedule only for high school classes. However, teacher can work in basic school to. However, in Lithuanian schools schedule creating starts from high school classes schedule. “School schedule optimization” program is working same way.

8 Optimization in Commercial Software

We discuss optimization possibilities of the following three commercial timetabling systems currently used in Lithuanian high schools: “Mimosa 2009”, “aSc TimeTables 2009”, and “Rector 2009”.

“Mimosa 2009” [18] is the product of the Finnish company “Mimosa Software Ltd”. “Mimosa” provides convenient GUI for manual timetabling and reports constraints violations.

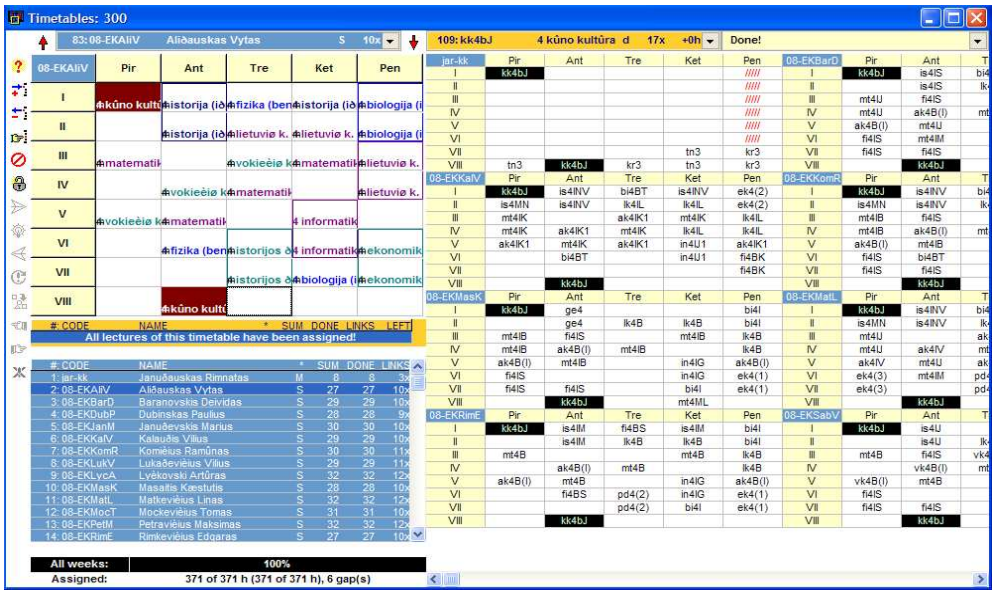


Figure 8. A fragment of “Mimosa 2009” output

Figure 8 shows a fragment of the output. In the upper-left side we can see pupils schedule, under it – pupils of the subject-group and in the right side – individual schedules of every pupil in the subject-group. The form is acceptable for Lithuanian schools. For example, “Ch3BK” means a chemistry lessons, pupils from 3-rd level, will learn as basic course. Optimization is limited to closing some gaps in teacher’s schedules. The software is popular in basic schools. Application in upper classes of high schools is possible within some strict limitations by setting individual pupil schedules. Long and hard manual work is needed if the school is large. Any penalty points are calculated in this program.

To compare results of different automatic optimization methods we need procedures for evaluation of undesirable factors in some fixed scales. In this paper, it is done in the framework of Pareto optimality [16]. The commercial software does not support this, since no direct comparison of decisions quality cannot be made.

"Rector 2009" [24] is the product of the Russian company "P. Yu. Smykalov". Figure 9 shows a fragment of output in the format similar to MS "Excel" forms used in local schools. In the upper side the subject for the group 12a are showed. Under it – all groups, lessons per week, subjects and teachers are showed. Green colour means, that no one works at the same time in two places. Reports, if one is trying to insert data to wrong place, are showed in red colour. Convenient for basic school scheduling. No automatic optimization.

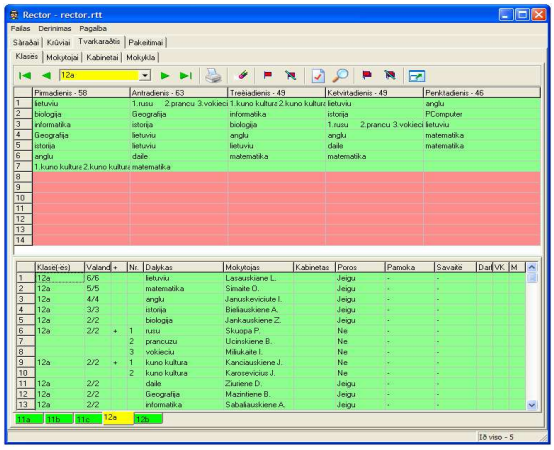


Figure 9. A fragment of "Rector 2009" output

Figure 10. A fragment of "aSc TimeTable 2009" output

"aSc TimeTables 2009" [3] is the product of the Slovak company "Applied Software Consultants s.r.o". A fragment of resulting timetable for Monday and Tuesday in a compact form for eight pupil subject-groups is in Figure 10. The results of experimental calculations are in Table 1. They show that the software works well in basic schools and is not practical in large high schools. Any penalty points are calculated.

Table 1. Testing „aSc TimeTables 2009"

		Small high school (50 pupils)						Medium high school (150 pupils)						Large high school (350 pupils)					
Settings of the program	**Complexity**	Small	Average	Hard	Small	Average	Hard	Small	Average	Hard	Small	Average	Hard	Small	Average	Hard	Small	Average	Hard
	Restrictions	Soft	Soft	Soft	Strict	Strict	Strict	Soft	Soft	Soft	Strict	Strict	Strict	Soft	Soft	Soft	Strict	Strict	Strict
Calculations	**Viewed options**	142800	183265	1652362	1375124	5026351	70054852	975236	4523625	---	33245895	---	---	---	---	---	---	---	---
	Left subjects	31	25	29	32	27	33	52	45	---	52	---	---	---	---	---	---	---	---
	Time	00:08:40	00:13:02	02:10:07	00:41:32	05:53:13	71:23:51	---	---	---	---	---	---	---	---	---	---	---	---

A timetable that satisfies all necessary conditions is regarded as feasible. A feasible timetable is optimal if it minimizes all undesirable factors. To compare the quality of different feasible timetables we must evaluate at least the most important undesirable factors. The difficulty is that desirability is subjective by definition and depends on the local conditions. This prevents comparison of results obtained by automatic optimization with decisions made by human operator.

9 Concluding Remarks

- The new element of this work is application and systematic investigation of the Bayesian Heuristic Approach (BHA) [20] to optimization of heuristic parameters (with penalty points). These include the initial temperature and the cooling rate of SA algorithm and the randomization parameter of the local search algorithm.

- BHA is intended for global optimization of functions with noise what is typical in optimization of heuristic parameters.
- The formulation of the objective function in terms of Pareto optimality seems to be new in the field of school scheduling.
- Application in some large schools shows some advantages comparing with commercial software. The web-site: http://soften.ktu.lt/~mockus and accompanying web-sites include corresponding.

References

[1] **Aarts, E.H.L.; Van Laarhoven, P.J.M.** 1987. Simulated annealing: *Theory and applications.* D. Reidel, Dordrecht, The Netherlands.

[2] **Abramson, D**. 1991. Constructing school timetables using simulated annealing: Sequential and parallel algorithms. *Management Science*, 37:98-113.

[3] *aSc Timetables.* 2009. <http://www.asctimetables.com/>

[4] **Bardadym, V. A.** 2003. Computer-aided school and university timetabling: The new wave? *In Lecture notes in computer science:* Vol. 1153, Practice and Theory of Automated Timetabling, First International Conference, Selected papers, pages 22-45. Springer.

[5] **Burke, E. K.; Trick, M.** 2005. Practice and theory of automated timetabling V. *Lecture notes in computer science*, Vol. 3616. Springer, Berlin.

[6] **Burke, E. K.; Eckersley, A. J.; McCollum, B.; Petrovic, S.; Qu, R.** 2004. Analysing similarity in examination timetabling. *In Proceedings of the 5th International Conference on the Practice and Theory of Automated Timetabling*, pages 557-559. Springer.

[7] **Burke, E. K.; De Causmaecker, P.** 2003. Practice and theory of automated timetabling IV. *Lecture notes in computer science*, Vol. 2740. Springer, Berlin.

[8] **Burke, E. K.; Petrovic, S.** 2002. Recent research directions in automated timetabling. *Journal of Operational Research Society*., 140:266-280.

[9] **Burke, E. K.; Erben, W.** 2001. Practice and theory of automated timetabling III. *Lecture notes in computer science*, Vol. 2079. Springer, Berlin.

[10] **Burke, E. K.; Carter, M. W.** 1998. Practice and theory of automated timetabling II. *Lecture notes in computer science*, Vol. 1408. Springer, Berlin.

[11] **Burke, E. K.; Ross, P.** 1996. Practice and theory of automated timetabling. *Lecture notes in computer science*, Vol. 1153. Springer, Berlin.

[12] **Carter, M. W.; Laporte, G.** 1998. Recent developments in practical course timetabling. *In Lecture notes in computer science:* Vol. 1408. Practice and theory of automated timetabling, pages 3-19. Berlin, Springer.

[13] **Carter, M. W.; Laporte, G.** 1996. Recent developments in practical examination timetabling. *In Lecture notes in computer science:* Vol. 1153. Practice and theory of automated timetabling, pages 3-21. Springer.

[14] **Dascalki, S.; Birbas, T.; Housos, E.** 2004. An integer programming formulation for a case study in university timetabling. *European Journal of Operational Research*, 153:117-135.

[15] **De Werra, D.** 1985. An introduction to timetabling. *European Journal of Operational Research*, 19:151-162.

[16] **Fudenberg, D.; Tirole, J.** 1983. Game Theory. *MIT Press*, Boston.

[17] **Metropolis, N.; Rosenbluth, A. W.; Rosenbluth, M. N.; Teller, A. H.; Teller, E.** 1953. Equations of state calculation by fast computing machines. *Journal of Chemical Physics*, 21:1087-1092.

[18] *Mimosa scheduling software.* 2009. <http://www.mimosasoftware.com/>

[19] **Mockus, J.** 2002. Bayesian heuristic approach to global optimization and examples. *Journal of Global Optimization*, 22:191-203.

[20] **Mockus, J.; Eddy, W.; Mockus, A.; Mockus, L.; Reklaitis, G.** 1997. Bayesian Heuristic Approach to Discrete and Global Optimization. *Kluwer Academic Publishers*, ISBN 0-7923-4327-1, Dordrecht-London-Boston.

[21] **Pedroso, J. P.; Moreira, N.; Reis, R.** 2004. A web-based system for multi-agent interactive timetabling. *In ICKEDS'04: International Conference on Knowledge Engineering and Decision Support*, pages 1-6, Porto, Portugal.

[22] **Pupeikienė, L.; Mockus, J.** 2005. School schedule optimization program. *Information Technology and Control*, 34:161-170.

[23] **Schaerf, A.** 1995. A survey of automated timetabling, technical report cs-r 9567. *Technical report*, Centrum Wiskunde & Informatica (CWI), Amsterdam, The Netherlands.

[24] **Smykalov, P**. 2009. School timetabling: Rector. <http://www.rector.spb.ru/uk/index.html>

[25] *Watt.* 2009. <http://www.asap.cs.nott.ac.uk/watt/>

[26] **Wren A.** 1996. Scheduling, timetabling and roistering a special relationship? *In Lecture notes in computer science:* Vol. 1153. Practice and theory of automated timetabling, pages 46-75. Berlin, Springer.