Application of Stateflow Diagrams in Production Line Modeling

Ján Čabala[*], Ján Jadlovský[**]

[*] Department of Cybernetics and Artificial Intelligence, Faculty of Electrical Engineering and Informatics, Technical University of Košice, Slovak Republic

[**] Department of Cybernetics and Artificial Intelligence, Faculty of Electrical Engineering and Informatics, Technical University of Košice, Slovak Republic

jan.cabala@tuke.sk, jan.jadlovsky@tuke.sk

***Abstract* - Paper aims on description of the Stateflow model of flexible assembly line. Assembly line is placed in Department of Cybernetics and Artificial Intelligence in Technical University in Košice. Stateflow model of this assembly line simulates its functionality and states, which occurs in production process. Outputs of this model are displayed in the last part of the paper. They can be used for various purposes, but mostly they will be used as data source for tasks dealing with time optimization or multi-objective optimization of production process.**

Keywords — **assembly line, simulation model, Stateflow, conveyor, post, production.**

I. Introduction

Automated production lines are the first choice in production process realization nowadays. Besides many other advantages, minimum or none of human intervention is needed during production process. This fact reduces possibility of failure caused by human factor. Results of many different advantages of these production lines can be named as costs minimization and minimization of time necessary for product construction.

Before the production line is put to real operation, it has to be designed in details. After this process, simulation of all subsystems, states and transitions is taking place in order to find all critical spots and possible failures. Simulation is cheaper and faster variant than building real prototype of assembly line. That is why simulation models are used for testing most of parameters of production process. All simulations are realized mostly before constructing the real production line or before putting the production line into real operation.

State model of this production line is realized via Stateflow diagrams, which are included in Simulink tool of Matlab software. These diagrams are used for creating and realizing simulations of discrete-event processes. Stateflow has also tools for simulation of parallel processing. This feature is necessary for creating this production line model, because many production sub-processes are executed at the same time. In first part of this paper flexible assembly line is described, followed by detail description of Stateflow diagrams of all posts contained in this assembly line. In the last part, results from simulation model are compared with data from real assembly line.

II. Flexible Assembly line

Flexible assembly line is model of automated production line placed in Laboratory of assembly lines and image recognition, owned by Department of Cybernetics and Artificial Intelligence in Technical University in Košice.

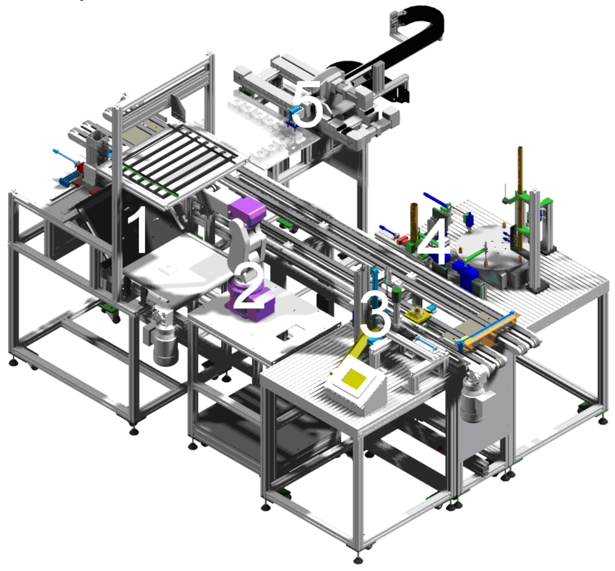

Figure 1. Flexible assembly line

This production line is built on mass production principles. In every post, different parts of final product are put on the palette. After finishing of all tasks in particular post, product is transported to another one by conveyor belt. The final product of this line can be named as specifically covered bearing.

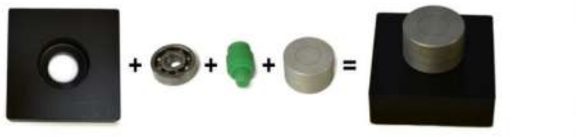

Figure 2. Product of flexible assembly line

Line consists of 5 production posts and special post, which ensures the palette transport between particular posts. Production process starts on post 1, where the base is released into production process randomly. After the release, base slides on the slope and finally falls on the conveyor. Post 2 is represented by robotic arm. After the image recognition process realized by camera fixed above the conveyor, robotic arm can detect the position and the angle of the base. Then the arm moves above the base, catches it and puts the base on the palette. Palettes are situated on the one of two main conveyors. On Post 3 bearing is put into the base. On post 4, shaft and cover are prepared on the rotary table. These parts are put into the bearing by pneumatic manipulator. After this process, the production process is finished, because post 5 operates with final product and ensures its transport from conveyor to warehouse into previously defined position. [1],[2] ,[3],[4].

III. Simulation model of flexible assembly line

Purpose of building simulation model is to simulate the production process in details and also to find most of mistakes from design phase. Some theoretical fundamentals for qualitative and quantitative system modeling are written in [5] .

Created simulation model is realized in Simulink programming interface. Functionality of flexible assembly line is simulated by means of Stateflow diagrams. Other possibilities of modeling discrete-event systems can be found in [6].Stateflow was chosen as modeling tool due to:

- its robustness in modeling discrete-event systems,
- its possibility of using Simulink and SimEvent blocks,
- previous positive experience with using MATLAB interface (using MATLAB in modeling logistic systems is described in [7]).

Stateflow diagram of this assembly line contains blocks, which represent particular posts, as well as blocks simulating transport of palettes between posts. Input parameters for this model are taken from Matlab interface. All of these parameters are created in initializing function, which is executed with the start of the simulation. Input parameters for this model are:

- number of desired products,
- type of bearing,
- type of shaft,
- type of cover,
- release time of base on 1,
- release angle of base on 1,
- initial status of bearing, shaft and cover stacks,
- count of realized simulations.

Release time and angle of the base could be generated also directly in Stateflow model, but Stateflow does not support Matlab commands for specification of random number generator algorithm. Since default generator generates always the same sequence of numbers, these parameters are created in initializing function and Stateflow considers them as constants. Output parameters of model are:

- execution times of particular posts,
- final execution time for each product,
- status of warehouse.

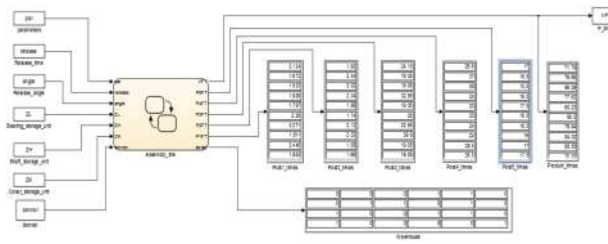
Figure 3. Assembly line model with inputs and outputs

In Figure 3. ,the model of production line in Simulink is depicted. Whole functionality of assembly line is placed in Stateflow block ASSEMBLY_LINE. On the left side are displayed inputs for the model, on the right side are depicted outputs generated by the model.

After entering the main Stateflow block, the Stateflow model of assembly line is displayed. Model consists of initialization part, where all input variables necessary for correct model functionality are set. Another part of model is functional part, where all processes running at assembly line are modeled by means of Stateflow diagrams. This part consists of two blocks running in parallel. One of them represents the conveyor and is used for simulating the palette transport. Other one represents particular posts of assembly line. All states of both main blocks can run in parallel, but states in these sub-states are exclusive (one post could be only in one state in time).

A. Post 0 – Palette transport

Transport of palettes between all posts in Stateflow model runs in Stateflow block CONVEYOR, which is running in parallel with block POSTS, where all activities realized in particular posts are modeled. Block CONVEYOR consists of 4 blocks running in parallel. These blocks control the palette transport between 4 stands, where palettes are either waiting for loading some part of final product (post 1-2,post 3, post 4) or waiting for placing the final product into the warehouse(post 5).

Realization of these 4 blocks is analogical. Each of these blocks contains only exclusive states. In first state, particular post is waiting for signal from previous post, which indicates finish of all previous tasks. This signal initializes transport of palette to another stand. Model is also checking presence of palette in particular post, so there is not a possibility of placing 2 palettes in same post, what would cause a collision and breaking the production process.

After successful evaluation of both input conditions (finish of particular post and free stand in next post), palette is after specific time (time of palette transport between previous and following post) coming to state, when it is ready for realizing all production tasks of particular post. After finishing all the tasks and checking the presence of palette on following post (if it is possible to move the palette to following post), palette is sent to further production process, until the final product is put into the warehouse on post 5.

As example of these 4 analogical Stateflow blocks, block providing transport between posts 3 and 4 can be seen on Figure 4.

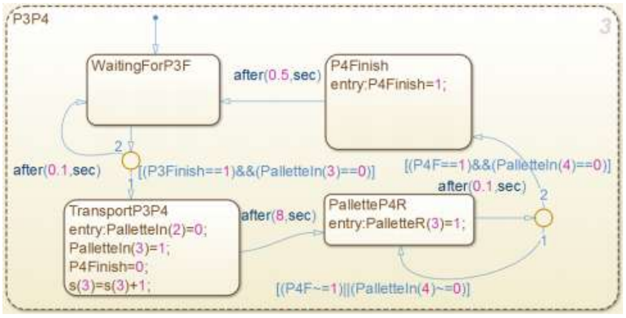

Figure 4. Palette transport from post 3 to post 4

B. *Post 1 – releasing the base*

Operation of post 1 in production line starts right after the start of the production process. The only action done in post 1 is releasing the base from its stack. This process is done randomly. Base is then sliding down and finally falls on the conveyor. All another actions dealing with transferring the base from conveyor onto the palette are realized in post 2. After releasing the palette, post 1 comes to state BASE_ON_CONVEYOR, which indicates finish of all processes realized by post 1.Then, post 1 is waiting for finishing the actions on post 2, because only one base should be laid on conveyor in Post 2 at the same time. If sufficient number of bases is released, post1 comes to state P1STOP, what means end of production process on post 1.

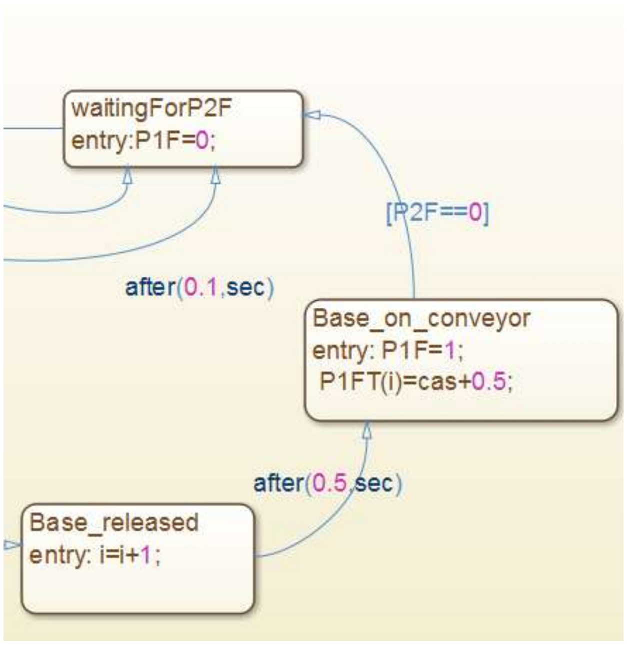

Figure 5. Fragment of Post 1Stateflow diagram

C. *Post 2 – robotic arm*

Post 2 is represented by robotic arm, which moves the base from moving conveyor onto the palette. At first, Post 2 is waiting for signal indicating finish of post 1. Subsequently, position and angle of the base are detected by camera system placed above the conveyor (state POSITION_DETECTED). In another state, robot moves above the base and it catches the base after short time (BASE_CAUGHT). After that, model is checking if the empty palette is ready on proper position. If not, the arm is waiting for the palette and checking the position of the palette in the loop. If the palette is ready for loading, robotic arm moves the base above the palette and put it on. Post 2 also includes the function SHAKE(), which simulates gentle movements with the base, which are done in order to proper base loading.

If the palette is placed properly, operation of post 2 is finished, palette with base is sent to post 3 and post 2 comes to state WAITING_FORP1. In this state it waits for another signal indicating readiness of another base released by post 1.After expedition of desired amount of loaded palettes, work on post2 stops and state P2STOP is activated.

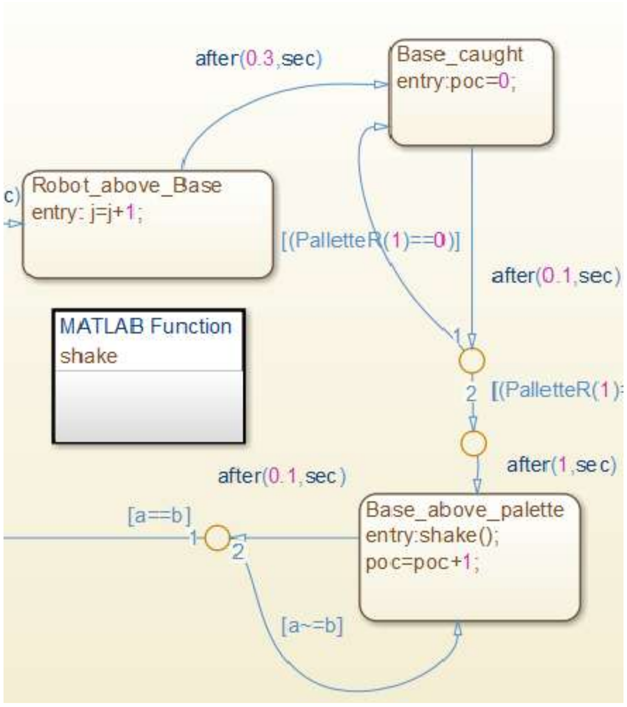

Figure 6. Fragment of Post 2 Stateflow diagram

D. *Post 3 – placing the bearing*

Post 3 is responsible for placing the bearing into the base. Since the process of preparing the bearing doesn't depend on finishing of any previous posts, post 3 starts right after the start of the production line. First step done in post 3 is pulling the bearing from the stack (state BEARING_PULL). In the stack, two types of bearings are randomly laid (large and small). Subsequently, the size of bearing is measured (BEARING_MEASURE). If the bearing has proper size, it is ready and it is waiting for the palette with base from the post 2. If the bearing is not corresponding with the order, it is removed from the production process and whole preparing of the bearing is done again, as far as right-sized bearing is found. Since only 10 bearings can be filled into the stack, after 10 iterations of bearing preparing process, stack must be refilled by the operator. This process is modeled by the state MECHANICAL_REFILLING, which lasts for 15 seconds, and then the production process goes on.

If all activities dealing with preparing the bearing are finished and palette from post 2 has already arrived, right bearing is caught and placed into the base by the manipulator. For all of these processes, corresponding states in Stateflow diagram are created.

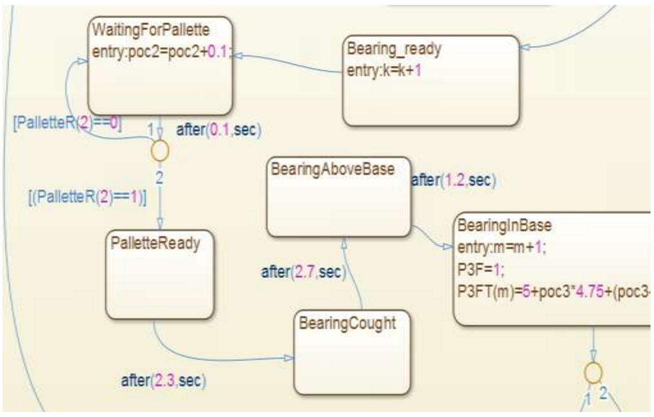

Figure 7. Fragment of Post 3Stateflow diagram

After finishing all its processes, post 3 is checking whether position for palette on post 4 is free (if so, palette is moved to post 4, otherwise it waits for finish of the production process on post 4), and at the same time it sends the signal about finishing its activities to post 2. Post 2 can then transfer the palette to post 3, in case it has already prepared another palette with base.

E. Post 4 – placing the shaft and cover

Post 4 is the last post of this production line, which is taking part in building the final product. In this post, the product is completed by putting the shaft into the bearing and covering the product with the cover. This post is represented by rotary table with 6 positions:

- position 1 - ejects shaft,
- position 2 - controls height of shaft,
- position 3 - controls color of shaft,
- position 4 – rejects shaft if it is different from order,
- position 5 - ejects cover (if shaft is on position 5 ,it is going automatically to position 6),
- position 6 - from this position pneumatic arm take shaft and put cover on bearing (if arm takes a shaft from position 6, table moves by one step)

Initialization of this post is similar to initialization of post 3 – it doesn´t depend on the previous posts and it starts immediately after starting the production process. According to the high complexity and parallelism of event occurred in this post, 2 parallel Stateflow blocks communicating with each other are created to simulate functionality of this post: block POST4_PREPARE for preparing the shaft and cover, and block POST4_REALIZATION responsible for placing the shaft and cover into the bearing.

Tasks in post 4 start with releasing the shaft from stack (state SHAFT_RELEASED). Subsequently, controls of height and color of the shaft are executed on position 2 and 3 of the rotary table. If any of controls find difference between the shaft and the production requirements, at position 4 shaft is rejected and taken away from the table (state WRONG SHAFT). If shaft is ready, table turns for 2 positions. At the same time it sends a signal to block POST4_REALIZATION. In case, there is a palette ready on the stand, pneumatic manipulator can catch the shaft from position 6 and place it into the bearing.

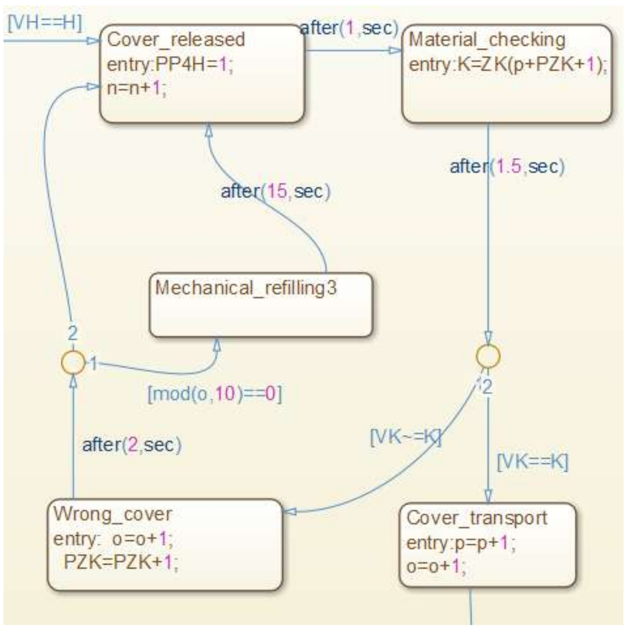

Figure 8. Fragment of Post 4Stateflow diagram – cover preparation

Signal indicating the correctness of the shaft is also the impulse for releasing the cover from its stack. Sensor checks the material of the cover. At the base of result of this control, cover is either removed from the production process, or it is placed on the position 5 of the rotary table. After putting the shaft into the bearing, rotary table turns and pneumatic manipulator puts cover on the shaft. After this event, product of this assembly line is ready to be sent into the warehouse.

Preparing parts for another product in this post can start only when all processes dealing with previous product are finished, because rotary table is not able to handle the preparing of two products simultaneously.

Simulation of refilling both stacks in this post is analogical to refilling the bearing stack in post 3.

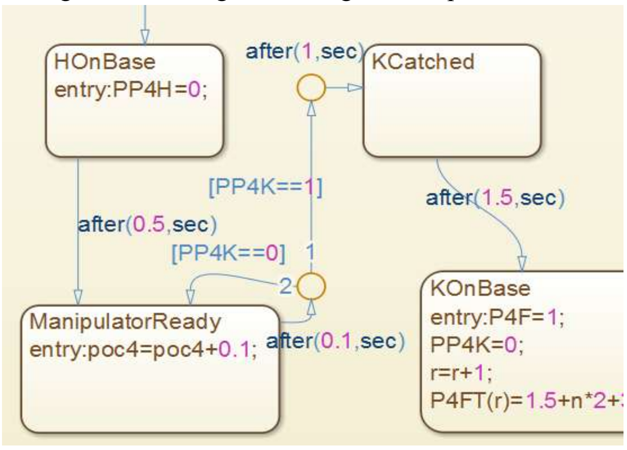

Figure 9. Fragment of Post 4Stateflow diagram – finishing the product

F. Post 5 – putting the product into the warehouse

Post 5 is the last post of this assembly line. Its purpose is to transfer the final product from conveyor to the chosen position in the warehouse. Product position in the warehouse is chosen in the function CHOOSING_PLACE(),which looks for the free position, where manipulator can place the product. In this function, the time needed for palette transport is counted as well. This time depends on the length of trajectory, which manipulator has to cross with palette from conveyor to the chosen position. After this transfer, all processes on the assembly line are finished. Post 5 then waits for sending another palette from post 4. After placing all products, production process ends and all posts of the assembly line are in their STOP states.

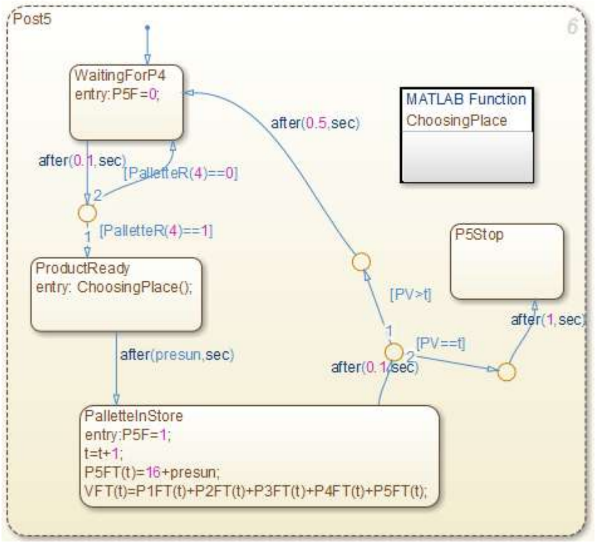

Figure 10. Stateflow diagram of post 5

IV. Comparison of simulation and real model

To compare the results from created simulation model and real assembly line model, 10 products were made in real model. Its average production time was 72,16 seconds. 50 simulations on simulation model were realized to demonstrate the similarity between models. In every simulation, 10 products were made and their average production time was calculated. Maximum difference between simulation and real line was 6 seconds. These results depict the similarity, although another work on this model will be done to minimize the differences between simulation and real assembly line model.

Results of the comparison can be seen on graph on Figure 11.

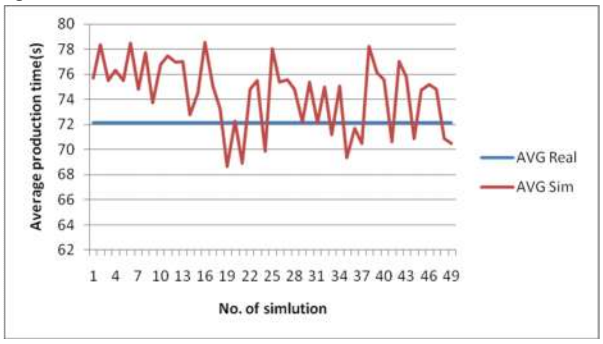

Figure 11. Comparing simulation results with real model

V. Conclusion

Created simulation model is simulating whole production process on assembly line, from start of production to the putting the last product to the warehouse. Output parameters of this model are production times of particular posts, as well as final production time of all products.

These parameters can be used as data source for production process optimization, as well as for solving multi-objective optimization tasks. Data from this model can be helpful in definition of mathematical model of production line, or optimization realized by evolutionary algorithms.

Another outputs can be added into this model, f.e. percentage of using particular posts can be watched. There can be also a possibility of finding critical spots of assembly line, idle times in posts and other factors, which can be helpful in further optimization of this assembly line model. Closer information about multi-objective optimization methods and its connection to assembly lines can be found out in [8]and[9], optimization methods using means of artificial intelligence are described in [10] .

Acknowledgment

This work has been supported by the Research and Development Operational Program for project: University Science Park Technicom for innovative applications with knowledge technology support, ITMS code 26220220182, co-financed by the ERDF (70%) and by grant KEGA - 001TUKE-4/2015 (30%) sponsor acknowledgments in the unnumbered footnote on the first page.

References

[1] JADLOVSKÝ, I.: Design and realization of control of flexible assembly line, Diploma thesis , KKUI FEI TU Košice 2012.

[2] VALIK, L.: Diagnostics of flexible assembly line using control program, Diploma thesis , KKUI FEI TU Košice 2015.

[3] JADLOVSKÝ, J. - ČOPÍK, M. - PAPCUN, P.: Distributed control systems. 1st release, Košice: elfa, 2013, 215 s., ISBN 978-80-8086-227-5 (in Slovak).

[4] ILKOVIČ, J. - PAPCUN, P.: Material flow in Flexible Assembly Company. In: SCYR 2012 : 12th Scientific Conference of Young Researchers of Faculty of Electrical Engineering and Informatics Technical University of Košice: proc. - Košice: FEI TU, 2012 S. 177-180. - ISBN 978-80-553-0943-9.

[5] KIM, J., & GERSHWIN, S. B.. Integrated quality and quantity modeling of a production line. OR spectrum, 27(2-3), 287-314, 2005.

[6] ČOPÍK ,M. :Methodics of design and realization of production line with focus on optimizing the production time, Dissertation thesis, KKUI FEI TUKE, 2014.

[7] BRUMERČÍK, F.: Discrete event simulation of logistic and transport systems. LOGI: Scientific Journal on Transport and Logistics, 2011,2.1:5-10.

[8] HRUBINA, K. - JADLOVSKÁ A. – HREHOVÁ S.: Algorithms of optimization methods with using programming systems. Vol.1 . Prešov – Košice, 2005.393 pages. ISBN 80-88941-31-8.

[9] ČABALA, J.: Multi-objective optimization of modern assembly lines. In: SCYR 2015: 15th Scientific Conference of Young Researchers : proceedings from conference : May 19th, 2015, Herľany, Slovakia. - Košice : TU, 2015 S. 250-253. - ISBN 978-80-553-2130-1 .

[10] REKIEK, B., et al. State of art of optimization methods for assembly line design. Annual Reviews in Control, 2002, 26.2: 163-174.