DEVELOPMENT OF GENETIC ALGORITHM FOR SOLVING SCHEDULING TASKS IN FMS WITH COLOURED PETRI NETS

Liana Napalkova, Galina Merkuryeva
Department of Modelling and Simulation
Riga Technical University
1/4 Meza str., LV-1048, Riga, Latvia
Email: {liana, gm}@itl.rtu.lv

Miquel A. Piera
Systems Engineering and Automatic Control Group
Universitat Autonoma de Barcelona
Bellaterra, 08193, Barcelona, Spain
Email: miquelangel.piera@uab.es

KEYWORDS

Scheduling, Flexible Manufacturing System, Coloured Petri Nets, Genetic Algorithms.

ABSTRACT

The paper describes the algorithm, which is developed to solve scheduling tasks in Flexible Manufacturing Systems. The algorithm is a combination of Genetic Algorithm and Coloured Petri Nets. It is proposed to use Coloured Petri Nets to tackle the encoding problem in Genetic Algorithm. The objective is to minimize the total make-span subject to different constraints obtained in Flexible Manufacturing Systems.

INTRODUCTION

The modern manufacturing environment is characterized by rapid product innovation, turnover and depreciation (Veen-Dirks 1998). Companies have to be more flexible in their operations to respond to product diversity and random fluctuations in the demand pattern. In accordance with the new market conditions, mass manufacturing systems have been replaced by Flexible Manufacturing Systems (FMS) with a low product volume and high flexibility (Reddy and Subramaniam 2003).

FMS are highly automated systems, which combine the benefits of automated transport systems and flexibility of job shop to produce various parts on a group of workstations connected by an automated material handling system. The ability of FMS to switch from the machining of one component to another without undue interruption enables products to be produced in small or medium quantities at very low costs.

However, FMS hardware flexibility makes it difficult to find an efficient way to achieve software flexibility. The reason lies in scheduling and dispatching multiple products through various processing routes, transporting them around the system and recovering from any failures in system components. The main complexity of solving FMS optimization problems is related to the knowledge representation about all relationships existed in a system. These relationships are often uncertain, nonlinear and combinatorial that influence the system behaviour and its performance (Russo and Sasso 2005).

The optimization approaches based on mathematical techniques, such as branch and bound or dynamic programming, have well-developed theoretical background. However, they are not always able to find optimal solutions to large instances in a reasonable amount of computing time (Jones and Rabelo 1998). Simulation based optimization allows representing complex relationships of a system, but it is rarely used to tackle scheduling problems, where the great number of possibilities to evaluate through simulation complicates its practical application (Piera, Guash, Narciso and Riera 2003). The most important and popular ones are metaheuristics, such as Genetic Algorithms, Tabu Search, Simulated Annealing, etc. These metaheuristics are high-level strategies that are used to guide an underlying, problem specific heuristic to increase their performance (Blum and Rola 2003). However, metaheuristics often meet a problem of good structural representation, which is a vital component of solving any optimization problem (Mesghouni, Hammadi and Pierre 2004).

Therefore, there is a need of powerful approaches to specify and optimize complex logistic and manufacturing systems. The development of hybrid methods, which would combine benefits of several techniques, such as artificial intelligence, operations research, simulation, heuristics, etc, is still an open area.

THE PROBLEM STATEMENT

The general scheduling problem is to find a sequence, in which the jobs pass through the system resources, that is optimal with respect to some performance criterion. The goal of the scheduling problem is to optimize some objective function depending on the applicative domain at hand.

In this research, the FMS scheduling goal is formulated as the minimization of total make-span represented by the following objective function:

$$f(\overline{w}, \overline{p}) = \min\{w_i + z_i \mid i = 1,\ldots,n\}, \qquad (1)$$

where:

- w – starting time of task i,
- p – processor,

- *z* – execution time of task *i*,
- *n* – number of tasks.

The optimal value of the objective function must satisfy the following constraints:

- *Functional constraints* define the types of operations that a specific resource can perform,
- *Capacity constraints* restrict the number of jobs that a resource can process at once,
- *Availability constraints* specify the availability of resources (e.g. number of shifts available on a group of machines),
- *Precedence constraints* define the sequence of operations in the job,
- *Processing time constraints* specify the time needed to perform the operation,
- *Setup constraints* require that each machine be in the proper configuration before performing a particular task,
- *Time-bound constraints* define an earliest acceptable release date of the job and a due date by which ideally it should be delivered to a customer.

THE PROBLEM FORMALISATION BASED ON COLOURED PETRI NETS

Coloured Petri Nets (CPNs) are among the most convenient and user-friendly graphical and mathematical tools for studying and modelling discrete-event systems due to their ability to represent the precedence relations and structural interactions of random, concurrent and conflicting events, which are typical for the FMS environment (Riera, Piera and Guasch 2001).

CPNs consist of transitions and places, which can correspond to events and their conditions, respectively. The dynamic behaviour of a system is modelled by evaluation of the markings, which mean the distribution of tokens on places. In order to identify different data types, attributes called as token colours are attached to tokens (similar to entities in traditional simulation models).

CPN formalism allows the specification of a system at different abstraction levels dependant on the modelling objective. Thus, the high level of flexibility allows changing the system configuration without modifying the CPN structure.

The use of CPNs both as conceptual and simulation models provides an ability to combine them with different analytical and optimization methods. This feature provides a possibility to apply CPNs to tackling different problems in logistic and manufacturing systems.

EXPLORING THE SEARCH SPACE WITH GENETIC ALGORITHMS

Genetic Algorithms (GAs) are search procedures for solving optimization problems. They keep a population of encoded solutions, which represent potential behaviour of a system. This population of solutions defines a set of points in the correspondent search space (Lodree and Geiger). GAs can process a large amount of information in parallel way. It results in simultaneous exploring different search regions. The search proceeds by evaluating a number of generations. For each generation the fitter solutions are selected to form a new population. There are three main procedures to be performed, such as:

- Reproduction,
- Crossover,
- Mutation.

The reproduction procedure selects individuals for the next generation. The selection highlights promising areas of the search space and, thus, guides a search direction. The crossover is aimed to exchange the information contained in current chromosomes. For that, individuals are mated randomly in the mating pool, and each two are crossed to obtain two new chromosomes. The mutation maintains diversity of the population. It usually works on a single chromosome and creates another one through alteration of the value of a string position or exchange of the values of two string positions. Thus, the mutation generally prevents the GAs from falling into local extremes.

The following issues are commonly raised in GAs:

- Representation (encoding) of the problem,
- Reproduction strategy,
- Design of genetic operators,
- Evaluation of the fitness function.

THE PROPOSED APPROACH

In order to solve the problem of structural representation in GAs, CPNs are used here both as conceptual and simulation models of the optimized system. Thus, CPN could be interpreted as a marionette, which reflects complex relationships within the system, but GA plays a role of director, which uses strings (i.e. chromosomes) to change the configuration of CPN and evaluate results according to the objective function.

In order to implement this approach, the following structural elements have been introduced:

- Standard Components Library,
- Standard Paths Library.

The role of Standard Components Library (SCL) is to compose the CPN of basic *puzzle* elements, i.e. Standard Components (SCs), which describe independent functional relationships in the system. Each SC consists of one transition, which connects a concrete input place with all possible output places. Transition in SC is used to model different operations. The type of operation is dictated by the token colour. Modelling an operation corresponds to the transition firing. Pre-conditions are assigned to transitions in order to enable or disable some transition. Post-conditions are applied to update the state of a system after the transition firing. Updating process changes the distribution of tokens on SCs. Combining all SCs results in complete puzzle of the CPN.

Standard Paths Library (SPL) generated from SCL provides all possible sequences of operations needed to produce target details. Each Standard Path (SP) satisfies precedence and functional constraints.

The following two types of chromosomes have been developed to manage the CPNs

- Goal-based chromosomes,
- Priority chromosomes.

Goal-based chromosomes define the production sequence of target details. Each gene denotes the type of a detail. In case of large production quantity, details can be grouped according to some routing information.

Priority chromosomes depend on Goal-based chromosomes, and describe the processing sequence of pieces. Each gene assigns one SP to the piece.

The final solution, represented with Goal-based and Priority chromosomes, is visualized using a Gantt diagram.

STEPWISE DESCRIPTION OF THE ALGORITHM

The developed algorithm consists of the following main stages:

- Defining the Standard Components Library,
- Generating the Standard Paths Library,
- Setting the scheduling goal and objective function,
- Creating the population of chromosomes,
- Running the Genetic Algorithm,
- Executing the Coloured Petri Net,
- Evaluating the fitness function and defining the optimal schedule.

Defining the Standard Components Library

The purpose of this stage is to describe the modelled system with independent SCs. For that, all places must be assigned to system's devices, such as assembly machines, Computer Numerical Control (CNC) machines, stocks, etc.

Generating the Standard Paths Library

This stage is dedicated to the automatic generation of SPL from SCL. Its task is to uncover all possible paths, which could be used to route pieces within the system.

Setting the scheduling goal

In this paper, the scheduling goal is to produce a certain quantity of target details with the minimum make-span. For that, one should define pieces used to assemble each type of the target detail.

Creating the population of chromosomes

The task of Goal-based chromosomes population is to represent all possible production sequences. Priority chromosomes population tackles the assignment and routing problems by means of attaching SPs to pieces.

Running the Genetic Algorithm

The GA begins with the creation of an initial population. In order to produce a new population of chromosomes from the initial (or current) one, individual chromosomes should be copied according to their fitness values. This process called as the reproduction consists of the following steps:

- Selection,
- Crossover,
- Mutation,
- Replacement.

Selection

The key feature of the proposed algorithm is that chromosomes can provide either executable or non-executable schedules. Theoretically, a non-executable schedule can give the best solution after swapping some genes. Therefore, just a simple comparison of fitness functions has no sense, because non-executable chromosomes will always have lower make-span values than executable ones, and, thus, solutions will be incorrect. The introduced selection strategy is focused on the evaluation of a proportion between these two types of chromosomes. During the execution of the algorithm five different situations can occur, such as:

1. The whole population consists of executable chromosomes;
2. The whole population consists of non-executable chromosomes;
3. Number of executable chromosomes is bigger than the number of non-executable ones;
4. Number of non-executable chromosomes is bigger than the number of executable ones;
5. Number of executable chromosomes is equal to the number of non-executable ones.

In the first case, the roulette wheel selection is used to choose parents for further reproduction. Since candidate solutions with lower make-span are less likely to be eliminated, all executable chromosomes have approximately the same probability to be selected. In the second case, all chromosomes are processed individually without crossing with other ones. The main idea of the third, fourth and fifth case is that each pair of parents selected for further reproduction consists of one executable and one non-executable chromosome. In other words, for each non-executable chromosome an executable one is selected using the roulette wheel.

Crossover
The crossover is applied only to Priority chromosomes, in order to change the paths assignment to tokens. The crossover operator is designed to avoid incorrect paths assignments. For that, the paths exchange is performed between genes, which have the same type of tokens. It means that paths marked in the first chromosome can be replaced with paths from the second chromosome only if these paths are attached to the same token colour.

Mutation
The Swap mutation is used to change values of two genes in selected chromosomes.

Replacement
Children are used to replace their parents.

Executing the Coloured Petri Net

The aim of this stage is to execute all CPN transitions specified in Standard Paths of Priority chromosomes, in order to reach the goal introduced in Goal-based chromosomes.

Evaluating the fitness function and defining the optimal schedule

The optimal schedule has the best fitness function. In order to calculate its value, the timeline, where all operations are placed consequently, is transformed into the Gantt diagram with parallel operations.

EXPERIMENTATION

FMS consisted of three subsystems have been used in order to verify and demonstrate the suggested methodology (Piera 2005):

- Load / Unload station,
- Manufacturing units,
- Transport system.

This FMS is aimed to produce two different types of pieces: type 123 (4) and type 13 (5), which are formed by pieces 1, 2, 3 and 1, 3, respectively. Before the assembly operation, the following transformations must be performed with pieces:

- The piece of type 1 must be drilled through,
- The piece of type 2 must be drilled twice: one long and narrow to be inserted in the type 1 piece, and one shorter and wider to fit the type 3 piece.

The software aimed to support the proposed algorithm has been developed in Borland Delphi 7 using the Object Pascal programming language. The software includes two main blocks of modules referring to:

1. Specification of the Conceptual Model,
2. Integration of the Conceptual Model with GA.

Two tests for two different configurations of FMS are performed using the developed software.

In the Test 1, a single manufacturing unit 1 of the FMS is considered as an optimized system (Figure 1).

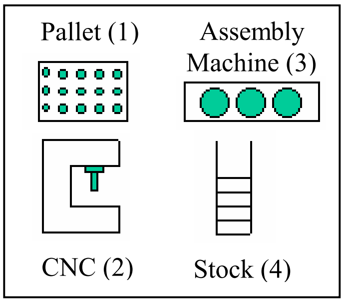


Figure 1. General scheme of manufacturing unit 1

This system can be described with four independent Standard Components (Figure 2).

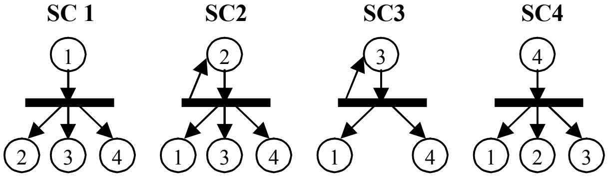


Figure 2. Standard Components Library

Standard Paths (Figure 3) are formed by combining these Standard Components in different ways.

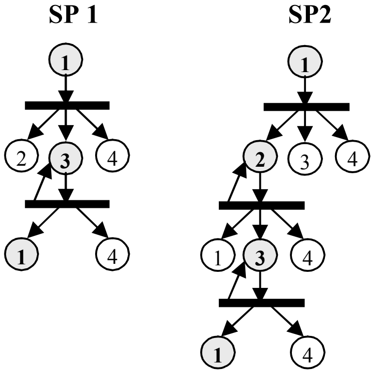
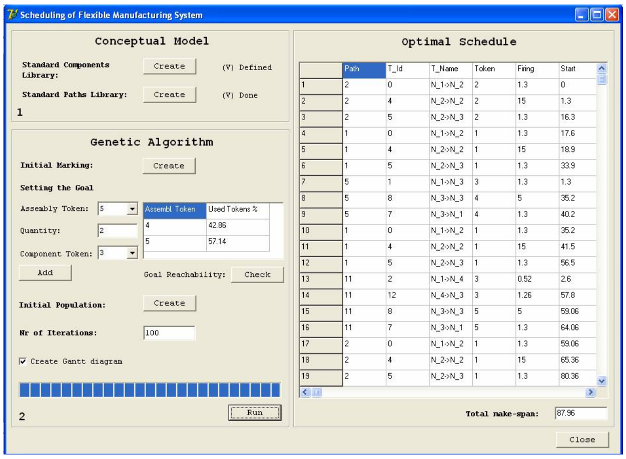


Figure 3. Sample Standard Paths

In this experiment, the scheduling objective is to assemble one piece of the type 4 and two pieces of the type 5 with the minimal make-span (Figure 4).



Standard Components Library: (V) Defined

Standard Paths Library: (V) Done

Initial Marking:

Setting the Goal

Assembly Token: 5

Quantity: 2

Component Token: 3

Assembl. Token	Used Tokens %
4	42.86
5	57.14

Goal Reachability:

Initial Population:

Nr of Iterations: 100

☑ Create Gantt diagram

	Path	T_Id	T_Name	Token	Firing	Start
1	2	0	N_1->N_2	2	1.3	0
2	2	4	N_2->N_2	2	15	1.3
3	2	5	N_2->N_3	2	1.3	16.3
4	1	0	N_1->N_2	1	1.3	17.6
5	1	4	N_2->N_2	1	15	18.9
6	1	5	N_2->N_3	1	1.3	33.9
7	5	1	N_1->N_3	3	1.3	1.3
8	5	8	N_3->N_3	4	5	35.2
9	5	7	N_3->N_1	4	1.3	40.2
10	1	0	N_1->N_2	1	1.3	35.2
11	1	4	N_2->N_2	1	15	41.5
12	1	5	N_2->N_3	1	1.3	56.5
13	11	2	N_1->N_4	3	0.52	2.6
14	11	12	N_4->N_3	3	1.26	57.8
15	11	8	N_3->N_3	5	5	59.06
16	11	7	N_3->N_1	5	1.3	64.06
17	2	0	N_1->N_2	1	1.3	59.06
18	2	4	N_2->N_2	1	15	65.36
19	2	5	N_2->N_3	1	1.3	80.36

Total make-span: 87.96

Figure 4. Running the GA for a single sub-system

The total make-span, achieved after running 100 iterations of GA, is equal to 87.96 minutes. Figure 5 illustrates the corresponding Gantt diagram.

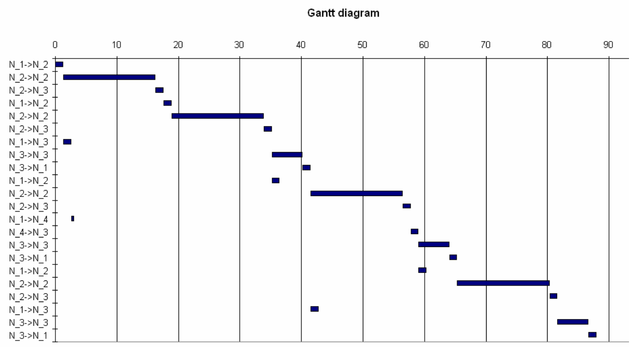


Figure 5. Gantt diagram for a single sub-system

In the Test 2, the manufacturing unit 1 and manufacturing unit 2 are combined to model more complex environment, which includes two concurrent sub-systems (Figure 6).

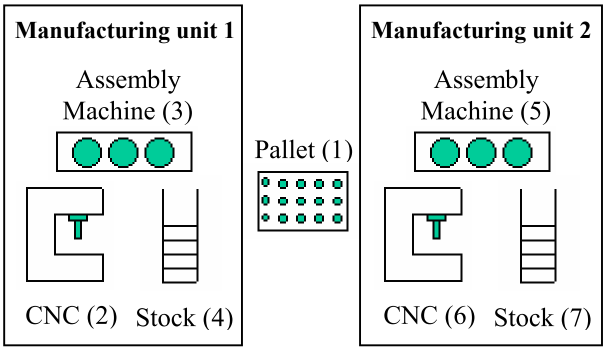
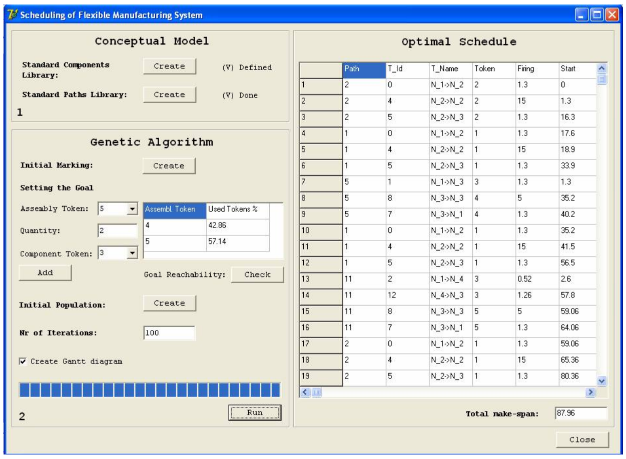


Figure 6. General scheme of manufacturing unit 1 and 2

The goal of this experiment is to assemble four pieces of the type 5 and, investigate a proportion of the work performed by two functionally similar sub-systems (Figure 7).

The total make-span received in 100 iterations is equal to 41.78 minutes.



Standard Components Library: (V) Defined

Standard Paths Library: (V) Done

Initial Marking:

Setting the Goal

Assembly Token: 5

Quantity: 2

Component Token: 3

Assembl. Token	Used Tokens %
4	42.86
5	57.14

Goal Reachability:

Initial Population:

Nr of Iterations: 100

☑ Create Gantt diagram

	Path	T_Id	T_Name	Token	Firing	Start
1	2	0	N_1->N_2	2	1.3	0
2	2	4	N_2->N_2	2	15	1.3
3	2	5	N_2->N_3	2	1.3	16.3
4	1	0	N_1->N_2	1	1.3	17.6
5	1	4	N_2->N_2	1	15	18.9
6	1	5	N_2->N_3	1	1.3	33.9
7	5	1	N_1->N_3	3	1.3	1.3
8	5	8	N_3->N_3	4	5	35.2
9	5	7	N_3->N_1	4	1.3	40.2
10	1	0	N_1->N_2	1	1.3	35.2
11	1	4	N_2->N_2	1	15	41.5
12	1	5	N_2->N_3	1	1.3	56.5
13	11	2	N_1->N_4	3	0.52	2.6
14	11	12	N_4->N_3	3	1.26	57.8
15	11	8	N_3->N_3	5	5	59.06
16	11	7	N_3->N_1	5	1.3	64.06
17	2	0	N_1->N_2	1	1.3	59.06
18	2	4	N_2->N_2	1	15	65.36
19	2	5	N_2->N_3	1	1.3	80.36

Total make-span: 87.96

Figure 7. Running the GA for two concurrent sub-systems

The Gantt diagram presented on Figure 8 shows that:

- Sub-systems work parallel to each other,
- Target details are produced in the proportion 50/50.

In other words, both the manufacturing unit 1 and manufacturing unit 2 assemble two details of the type 5.

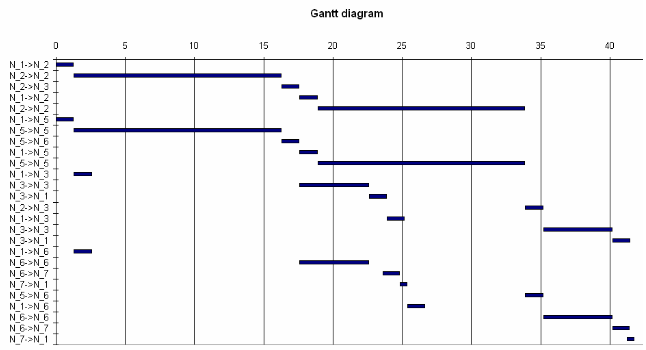


Figure 8. Gantt diagram for two concurrent sub-systems

These results seem to be optimal in the environment of concurrent units with similar configurations.

CONCLUSIONS

Paper introduces the algorithm aimed to solve scheduling tasks in Flexible Manufacturing Systems. It applies Coloured Petri Nets to tackle the encoding problem in Genetic Algorithms. In order to implement this approach, the decision-support tool has been developed using Borland Delphi 7 software. The comprehensive examination has been made to test the algorithm. First, the scheduling problem of a separate sub-system was solved. Second, an attempt to combine two concurrent sub-systems was made to demonstrate the applicability of the algorithm to the real-world FMS scheduling problems and encourage the further work. The results of this examination proved the efficiency of integrating Coloured Petri Nets within Genetic Algorithms. The most important result is that the algorithm correctly operates with two concurrent subsystems. It means that the proposed approach can be used to schedule consequent and parallel operations within a system. Moreover, it can be applied for scheduling jobs in systems that have more complex configurations.

However, the final results don't depend only on the good structural representation of GA. The reproduction strategy also plays a vital role in the GA performance. Currently, the selection procedure is based on division of a population into executable and non-executable chromosomes. Despite the fact that this procedure provided good results, additional experiments should be performed with more complex system configurations.

REFERENCES

Blum, C. and Rola, A. "Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison". In: *ACM Computing Surveys*, vol. 35, no. 3, pp. 268-308, 2003.

Jones, A. and Rabelo, L. C. "Survey of Job-Shop Scheduling Techniques", 1998.

Lodree, E. J. and Geiger, C. D. "A Simulation Optimization Approach for the Two-Product Newsvendor Problem".

Mesghouni, K., Hammadi, S. and Pierre, B. "Evolutionary Algorithms for Job-Shop Scheduling". In: *Int. J. Appl. Math. Comput. Sci*, vol. 14, no. 1, pp. 91-103, 2004.

Russo, M. F. and Sasso, A. "Modelling, Analysis, Simulation and Control of Laboratory Automation Systems Using Petri Nets: Part 1. Modelling". In: *JALA Tutorial*, vol. 10, pp. 172-181, 2005.

Piera, M. A., Guash, A., Narciso, M. and Riera, D. "Optimisation of Logistic and Manufacturing Systems through Simulation: A Coloured Petri Net Based Methodology", 2005.

Piera, M. A. "Production planning of a Flexible Manufacturing System", Universitat Autonoma de Barcelona, 2005.

Reddy, K. R. B. and Subramaniam, V. "Dynamic Scheduling of Flexible Manufacturing Systems", 2003.

Riera, D., Piera, M. A. and Guasch, A. "PN to CSP methodology: Dichotomic Search", 2001.

Veen-Dirks, P. "Implications of the new manufacturing environment for management control system design", 1998.

Visipkov, V. "Modern Metaheuristics for Combinatorial Problems of Logistics: A Survey of Applications". In: *Simulation: Applications, research and education in the Baltic Area*. Research report series nr. 2. Co-editor: Yuri Merkuryev, supported by Swedish Institute, Sweden, pp. 9-48, 1999.

AUTHOR BIOGRAPHIES

LIANA NAPALKOVA holds a MSc degree in Computer Science from Riga Technical University (2006). Currently, she is a research assistant at RTU Department of Modelling and Simulation, and participates in research projects in the logistics field.

GALINA MERKURYEVA is Professor at the Department of Modelling and Simulation, Riga Technical University. She holds Doctor of Engineering (Dr.sc.ing) from Institute of Electronics and Computer Techniques of Latvian Academy of Sciences and Doctor of Technical Sciences (Dr.tech.sc.) from Institute of Control Sciences of Russian Academy of Sciences. G. Merkuryeva has professional interests and experiences in discrete-event simulation, simulation metamodeling, decision support tools development, and simulation based training.

MIQUEL ANGEL PIERA received his MSc (Control Engineering) from the University of Manchester Institute of Technology in 1990 and his PhD degree from the Autonomous University of Barcelona (Spain) in 1994. He participates in industrial research projects in the logistics and manufacturing field and at present he is Co-director of LogSim, a Modelling and Simulation Institution sponsored and founded by the local government of Catalonia. Prof. Piera is the coordinator of the Spanish Simulation group of the International Federation of Automatic Control. Recently, he has published a modelling and simulation book that is being used for teaching in many Spanish universities.