DOI: 10.2478/v10051-010-0013-2

Development of a Web Application for Dynamic Production Scheduling in Small and Medium Enterprises

Davorin Kofjač[1], Andrej Knaflič[2], Miroljub Kljajić[1]

[1]University of Maribor, Faculty of Organizational Sciences, Kidričeva cesta 55a, SI-4000 Kranj, Slovenia, davorin.kofjac@fov.uni-mb.si, miroljub.kljajic@fov.uni-mb.si
[2]Mimovrste d.o.o., Spodnji plavž 24e, SI-4270 Jesenice, Slovenia, nejc@domenca.si

This article describes the development of a web-based dynamic job-shop scheduling system for small and medium enterprises. In large enterprises, scheduling is mainly performed with appropriate technology by human experts; many small and medium enterprises lack the resources to implement such a task. The main objective was to develop a cost-effective, efficient solution for job-shop scheduling in small and medium enterprises with an emphasis on accessibility, platform independence and ease of use. For these reasons, we decided to develop a web-based solution with the main emphasis on the development of an intelligent and dynamic user interface. The solution is built upon modular programming principles and enables dynamic scheduling on the basis of artificial intelligence, i.e. genetic algorithms. The solution has been developed as a standalone information system, which allows the management of virtually all scheduling activities through an administration panel. In addition, the solution covers the five main functionalities that completely support the scheduling process, i.e. making an inventory of resources available in the company, using it in the process of production planning, collecting data on production activities, distribution of up-to-date information and insight over events in the system.

Keywords: dynamic job-shop scheduling, genetic algorithms, web application development, web user interface, small and medium enterprises

1 Introduction

An increasing number of companies manufacture their products for each individual customer. Time, together with costs and quality of products, is becoming a crucial factor for a successful business. The time component is especially critical in a make-to-order production facility, where flexibility and reduction of process flow-time determine the rate of success of a company (Buchmeister, Palčič and Pavlinjek, 2008).

The ever-changing business environment is forcing small and medium enterprises (SMEs) to perform management tasks, i.e. production planning, in real-time to meet the delivery dates. Production plants that produce a large number of different specific products cannot hold them all in stock. SMEs also usually have limited production capabilities, which forces them into rash decisions regarding which orders to take from the customers when their production capabilities are exceeded. Such decisions are usually far from optimal, since the quality of their decisions should depend on the synergy effect of their intuition as well as on knowledge and skills, supported by tools that need to be directed into a creative systemic thinking and (co)-operation when solving complex problems (Kramberger and Rosi, 2007). The decision on whether to take another order from the customer is closely correlated to the existing production plan. Hence, if the production capacities are full, or the production plan is difficult to determine, SMEs face severe problems in meeting the delivery dates and in making quality decisions (Choi et al., 2004).

Production processes involve numerous operations with distinct technological ordering, where processing times have to be known as well the workplaces or machines where these operations have to be performed. With increasing numbers of operations and machines, production planners, who mostly use only their experience, intuition and simple techniques, encounter the problem of finding a “good” production schedule in an acceptable amount of time. The obvious consequence is inadequate machine utilization, exceeded project deadlines, increased costs and unsatisfied customers. SMEs are affected by this problem much more than large enterprises, which have the funds and other resources to tackle production-scheduling problems. In SMEs, the production plan is usually dependent on individuals or very small groups of experts that lack the support of modern information technology. Furthermore, customized production requires frequent production rescheduling as opposed to the production in which large series of products are manufactured. Frequent rescheduling is a consequence of many customers; it obstructs the production of large series, since each customer has its own requirements that the production must adapt to. However, this is the advantage of SMEs, because they produce unique goods with a greater added value.

Stawowy et al. (2007), and Derigs and Jenal (2005) concluded that it is impossible to implement a universal planning system that would cover the wide spectrum of different production branches. Each enterprise needs a tailored version of a planning system, because incorporating all specific requirements of each production branch into a single planning system is not reasonable. Hence, because of the soft and context-dependent aspects of each production branch, to have fully automatized planning processes is not to be expected.

The production planner not only requires production-planning software supported by algorithms, but also all the information available in order to make his decisions easier. Therefore, the user interface has to be intelligent and dynamic, enabling the user to integrate his own ideas and decisions, thus guiding the user through critical situations (Marinho et al., 1999). Lately, there have been an increased number of applications running over the internet. Internet solutions are less expensive, more effective and much easier to maintain than the classic installed software. Web-based applications support different platforms and are easily utilized by different business entities (Tarantilis et al., 2008).

Hence, our goal was to develop a cost effective, modern, robust, accessible and user-friendly web-based application to support production scheduling in SMEs with the use of artificial intelligence (genetic algorithms) as the backbone of the production-scheduling engine. The solution had to be accessible from anywhere, platform independent and easy to use.

2 Production planning and scheduling

Scheduling is a selection process between alternative plans, as well as a determination of the resources and times of each while taking resource constraints into account (Dorn and Slany, 1994). The production-scheduling problem arose in the early 20th century with the emergence of make-to-order production (Jain and Meeran, 1998). Scheduling problems are constantly increasing as development trends force make-to-order manufacturers into quick adaption to customers' demands and competition (Yen and Crow, 1998).

In general, production scheduling is a NP-hard problem that is solvable with robust methods over an infinite or a very long period. A production scheduling method is considered to be good if it reaches a near-optimal solution in a reasonable amount of time (Caseau and Laburthe, 1995).

Muth and Thompson (1963) were among the first who tried to engage the production-scheduling problem. Their model is still used as a quality test of modern scheduling methods. Their goal was to minimize the makespan, but there are many other measures of schedule quality, such as tardiness, etc. There are many methods for solving scheduling problems, such as Branch & Bound, Johnson's algorithm, etc.; however, Davis (1985) was the first to demonstrate the use of genetic algorithms (GAs) to solve scheduling problems. Recently, Kljajić et al. (2001) concluded that the use of GAs contributes to better production scheduling. They used a simulation model and GAs to achieve a better utilization of machines and a decrease of costs associated with it. Consequently, the throughput of the production line increased. Breskvar and Kljajić (2001) showed how to improve scheduling of large series of products with the interconnection of the simulation model and GA to achieve an automatic scheduling process.

In recent times, approaches to reduce high computational runtimes have been emerging. Kofjač and Kljajić (2008) developed a production-scheduling prototype with GA that is supported by a visual simulation model to validate the scheduling algorithm and to incorporate expert knowledge into the production scheduling process. Hence, the search space of GAs is considerably smaller, thus significantly improving the scheduling process runtime.

A key factor for a real-time scheduling process is up-to-date data regarding available manpower, broken machines, cancelled orders etc. It is known that updating data is possible in fully automatized processes. In contrast, keeping data up-to-date in processes where people are involved is a problem. In addition, keeping data up-to-date requires a significant amount of time if this process is not supported by information technology (Ljubič, 2000).

While larger enterprises can invest in adequate technology, such as ERP systems, to support their production processes, numerous SMEs have to cope with a lack of such support. Although ERP systems are complete information technology solutions, they require high maintenance and counselling costs (Na et al., 2003). Furthermore, one of the major drawbacks of ERP systems is their lack of web support, modularity and connectivity with other companies' information technology (Akkermans et al., 2003).

Make-to-order manufacturing cannot depend on standard production processes used for mass production, because make-to-order production usually involves different production activities for each customer's order. Furthermore, it is not reasonable to make a stock of such specific products. Hence, frequent production rescheduling is required to adapt to a new customer's order. In such cases, production activities have to focus on reducing overdue deliveries. Therefore, SMEs need a solution that enables the planning and scheduling of customers' orders while utilizing minimal computational power, which is also supported by web-based applications. Since most SMEs lack resources (manpower, funds, cost effective technology etc.) that would allow them to quickly adapt to customers' demands, there is a great need for a solution that would simplify and speed up the production planning process (Choi et al., 2004).

Several authors have researched the use of web applications in production environments. Qiu et al. (2001) proposed a distributed web multi-user environment for a predefined production plan. Ong et al. (2002) developed a web solution for production assessment in a milling industry, which supports fault detection and energy consumption prediction. Chung and Peng (2004) implemented an object-oriented web technology for management of complex machine and tool selection. Their web application is a complete solution, adequate for most SMEs with limited funds for investments.

In our case, if we wish to perform scheduling in the proposed web application, we would have to go through the scheduling process shown in Figure 1, which is presented generally. First, we have to define workplaces or machines (CNCs, table saws, etc.). Then we have to define operations (gluing, sawing, etc.) that, together with workplaces, define a job (front side of a drawer, top of the frame, etc.). A set of jobs defines the BOM (the bill of materials; in this case: top drawer, bottom drawer, left and right doors, etc.), while a set of BOMs (wardrobe, table, chair, etc.) defines a project, which represents an order placed by a customer. After all the projects are defined, scheduling can be performed and then evaluated through reporting. Usually, workplaces, operations, jobs and BOMs are defined only once, when the scheduling system is initialized and the data is imported. Afterwards, the scheduling process usually consists only of project definition for each customer, scheduling and report analysis. However, if, for example, a customer requires a cabinet that needs an additional drawer, one may to redefine the BOM and, consequently, the job and some operations. If one needs to update the workplaces, this is usually done when new equipment is bought or when a machine breaks down.

When the schedule has been calculated, it is output via reports. Designated users are allowed to view reports according to their user privileges, e.g. reports for specific workplaces, and perform production activities, such as marking the operation as complete, on a touchscreen in the production hall. In this way, feedback is provided instantly to the production planner so he can make appropriate decisions or maybe even reschedule the production plan to take new or high priority orders into account.

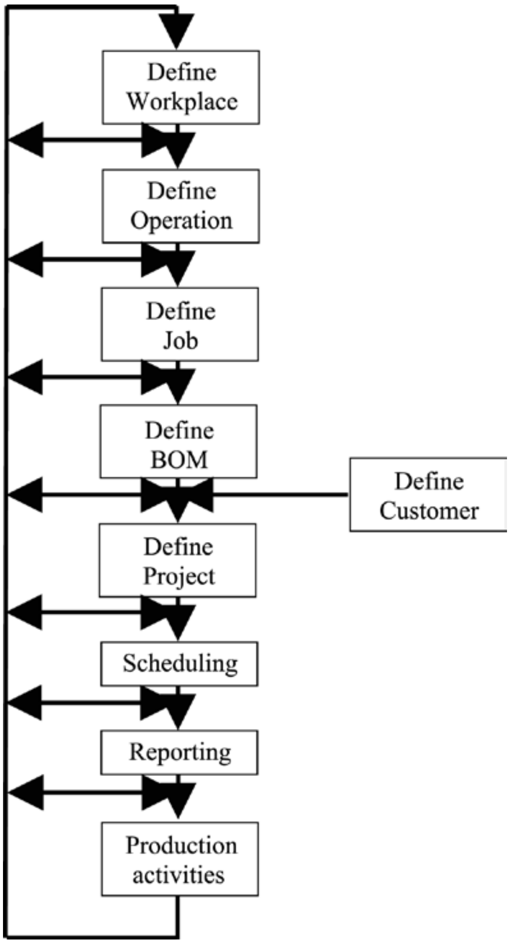


Figure 1: A generalization of a scheduling process.

Development of a production-scheduling algorithm

Scheduling is one of the key processes in production and plays a very important role in a make-to-order manufacturing, which requires great flexibility to meet the constantly changing customer demands. Therefore, the production process needs to be rescheduled frequently to achieve the optimal/ near-optimal schedule regarding the criteria function, which in our case is the shortest makespan.

While performing the scheduling process, we face two contradictory criteria: the makespan and the scheduling process runtime. Of course, we want to achieve the lowest makespan possible, but we also have to consider the scheduling process runtime in which the particular makespan is achieved. The scheduling process has to be performed in real-time in order to satisfy the solution feasibility. For example, if production rescheduling is required daily, it would be unacceptable for the scheduling process to take half a day to perform its calculations, regardless of whether it would achieve an optimal schedule. In such a case, we have to consider the near-optimal schedule that was calculated, e.g. in one hour or less.

Therefore, before the scheduling process is started, we have to optimize input parameters. If there are too many input parameters, the scheduling process runtime increases exponentially with the increasing number of input parameters. Furthermore, we also have to consider the simulation timestep Δt, which is an important factor while performing the scheduling process. If we select a Δt that is too short, the simulation runtime would increase greatly, but we would achieve a “fine grained” schedule that would provide much detail. The question is whether we need so much detail. For example, the furniture production does not need a Δt measured in seconds; it is sufficient that the Δt is measured in minutes. In contrast, some other production process, e.g. in the chemical industry, would probably require the Δt measured in seconds or even perhaps in milliseconds.

Job-shop scheduling

We are dealing with make-to-order furniture production, in which furniture is produced in very small series or no series at all. In our case, most of our customers’ orders are different. Therefore, production scheduling has to be performed frequently to ensure the optimal/near-optimal production schedule.

The base for scheduling is the bill of materials (BOM) needed to complete a product, e.g. a wardrobe. In this case, a BOM would include a frame, drawers, a door etc. In contrast, for example, a drawer’s BOM would include four sides, one bottom, sliders, and so on. Such a BOM can be represented by a tree structure as shown in Figure 2.

One can easily notice that certain parts have to be completed before the assembling of other parts can begin (precedence). Such a tree structure is basically a set of n jobs $J = \{ j_1, j_2, \ldots, j_n \}$ that have to be performed on a given set of m

machines $M = \{ m_1, m_2, \ldots, m_m \}$. A job j_i consists of a set of k operations $O_i = \{ o_{i1}, o_{i2}, \ldots, o_{ik} \}$ that are performed on a subset of machines $H \subseteq M$. An operation o_{ik} is defined by the uninterrupted time period t_{ik} needed to perform this operation and a machine h_{ik} on which the operation is performed. In our case, a job can be repeated on the same machine (recirculation). Furthermore, jobs have to be processed in a given sequence as shown in the example presented by a tree structure in Figure 2. The objective in our case is to produce a product in the shortest time possible, i.e. to minimize the makespan denoted by C_{max} that is defined as the time when the last job leaves the system:

$$\min C_{\max} = \max(C_1, C_2, ..C_n), \qquad (1)$$

where C_i is the completion time of job j_i.

The problem defined above can be described as the job-shop scheduling (JSS) problem, which is one of the best-known machine scheduling problems. Several techniques to solve complex JSS problems are described in Choi and Yang (2006). In our study, genetic algorithms were used to optimize the production schedule.

Solving job-shop scheduling problem by genetic algorithms

Genetic algorithms (GAs) can be used to search efficiently in a large search space, without explicitly requiring additional information (such as convexity, or availability of derivative information) about the objective function to be optimized (Naso et al., 2007). For this reason, in the previous decade, they have been applied to many (combinatorial) problems, including scheduling.

GA representation of the JSS problem

Recently, different GA representations for job-shop scheduling problems have been proposed, e.g. operation-based, job-based, machine-based, etc. (Gen and Cheng, 1997). In our research, we have used an operation-based representation that encodes a schedule as a sequence of operations, with each gene standing for one operation. All operations for a job are given with the same symbol and are then interpreted accordingly to the order of occurrence in the sequence for a given chromosome, as proposed by Gen, Tsujimura and Kubota (Gen and Cheng, 1997).

Consider the three-job four-machine problem given in Table 1. Jobs j_1 and j_2 have to be completed before the processing of job j_3 takes place. Hence, a chromosome is divided into sections (see Figure 3), and crossover and mutation can be performed only inside a section.

Table 1. An example of a three-job four-machine problem – operation machine sequence for each job.

	Operations			
Job	1	2	3	4
j1	m1	m3	-	-
j2	m1	m3	m4	-
j3	m1	m2	m1	m4

2	2	1	1	2	3	3	3	3
Section 1					Section 2			

Figure 3: A chromosome division into sections regarding the job precedence constraint.

Suppose a chromosome is given as [2 2 1 1 2 3 3 3 3], where 1 stands for job j_1, 2 for job j_2, and 3 for job j_3. Job j_1 has two operations (there are two 1's in the chromosome), job j_2 has three operations, and job j_3 has four operations. For example, the first 2 corresponds to the first operation of job j_2 (which will be processed on machine m_1), the second 2 corresponds to the second operation of job j_2 (which will be processed on machine m_3), and the third 2 corresponds to the third operation of job j_2 (which will be processed on machine m_2) as shown in Figure 4 (respectively for jobs j_1 and j_3). Note that Operations 1 and 3 of job j_3 are performed on machine m_1 (recirculation).

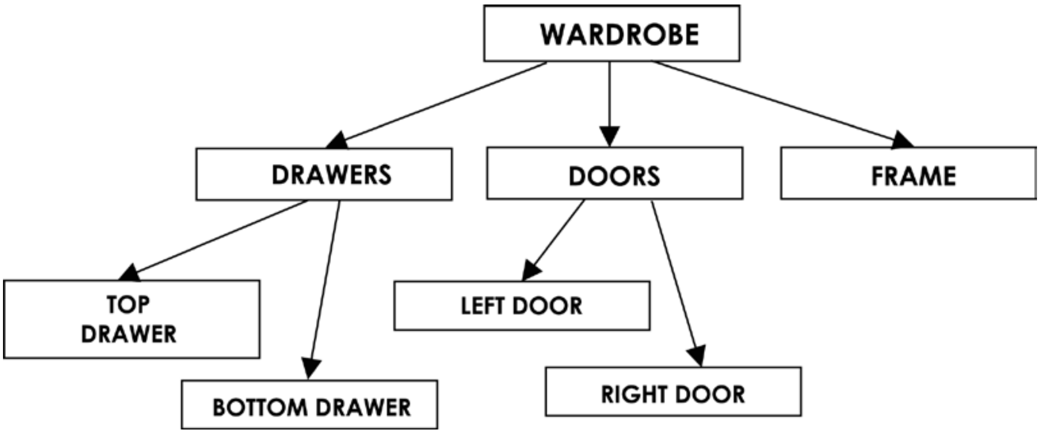


Figure 2: A part of a wardrobe BOM represented by a tree structure.

Initial population

The initial population of chromosomes was generated in two ways: randomly and by mirroring. Random generation is a commonly used technique for providing an initial population of chromosomes in the absence of *a priori* information about the solution. In the first step, *N* equal individuals are created as described earlier. Next, the genes in each section are moved to a random position inside the same section, thus providing a random schedule.

Because random generation does not necessarily provide a good initial population, the mirroring of initial population was used (Kofjač and Kljajić, 2008). Kofjač and Kljajić proved that it is possible to improve our chances of starting with a fitter solution by simultaneously checking the mirrored solution. By doing this, the fitter one (random or mirrored) can be chosen as an initial solution. Therefore, starting with the closer of the two guesses (as judged by its fitness) has the potential to accelerate convergence.

Mirroring is performed as follows: in the first step, an initial population of size *N* is generated randomly. Next, each individual in a randomly generated population is mirrored, i.e. each section inside an individual is mirrored, thus obtaining another *N* individuals. Individuals of both populations are then grouped and ranked according to the fitness function. Finally, the best *N*-ranked individuals are taken as the initial population. By utilizing mirrored points, we can obtain fitter initial population even when there is no *a priori* knowledge about the solution(s).

Chromosome:	2	2	1	1	2	3	3	3	3
Machine:			1	2					

(a)

Chromosome:	2	2	1	1	2	3	3	3	3
Machine:	1	2			4				

(b)

Chromosome:	2	2	1	1	2	3	3	3	3
Machine:						1	2	1	4

(c)

Figure 4: Operations of jobs and corresponding machines: (a) for job j1, (b) for job j2 and (c) for job j3.

Selection

Several selection methods to choose chromosomes for the next generation were tested: roulette selection (see Gen and Cheng, 1997), fitness rank distribution (Reeves, 1995), and Fibonacci selection (Bernik, 2001). Also, the elite selection is performed to preserve the best individuals.

The roulette selection is a fundamental selection method used in GA. In roulette selection, the fitness function assigns a fitness value to chromosomes. This fitness value is used to associate a probability of selection with each individual chromosome. If f_i is the fitness of an individual *i* in the population, its probability of being selected is:

$$p_i = \frac{f_i}{\sum_{j=1}^{N} f_j} \tag{2}$$

where *N* is the number of individuals in the population. While candidate solutions with a higher fitness will be less likely to be eliminated, there is still a chance that they may be.

The fitness rank distribution (FRD) was introduced by Reeves (1995). It selects parents according to the following probability distribution:

$$p_i = \frac{2i}{M(M+1)} \tag{3}$$

where *i* refers to the *i*-th chromosome in descending order of makespan and *M* refers to the fittest one. This implies that the median value has a chance of 1/*M* of being selected, while the fittest one (the *M*-th chromosome) has a chance of 2/(*M* + 1), roughly twice the median.

The Fibonacci selection was first introduced by Bernik (2001) and uses a Fibonacci sequence (0, 1, 1, 2, 3, 5, 8, etc.) to select individuals. The Fibonacci sequence is defined as follows:

$$F_n = \begin{cases} 0 & \text{if n} = 0; \\ 1 & \text{if n} = 1; \\ F_{n-1} + F_{n-2} & \text{if n} > 1. \end{cases} \tag{4}$$

Individuals from a population are selected according to the places defined by Fibonacci numbers. By applying this method, the fittest individuals are selected more often than the less fit ones. Hence, the variety of population is maintained and the survival of the fittest concept is being applied, while the GA converges faster towards the solution.

Crossover

To produce a new generation of chromosomes, a crossover is applied with a given probability (p_c). The one-cut-point crossover (CUT) and linear order crossover (LOX) were tested. Note that crossover can be applied only inside a section (e.g. Section 1).

Consider two parents and two child chromosomes for CUT crossover as shown in Figure 5. In the first step, genes [2 2] are copied from Parent 1 to Child 1 up to the crossover point. Respectively, genes [2 1] are copied from Parent 2 to Child 2. Child 1 is missing the genes [1 1 2] and Child 2 is missing the genes [2 1 2]. The missing genes gap is filled in Step 2. To produce a feasible schedule, the gap in each child must be filled with the missing genes by taking each legitimate gene from the other parent in order. For example, a gap in Child 1 is filled with the sequence [1 2 1], while the gap in Child 2 is filled with the sequence [2 2 1]. Note that the number of 1's and 2's in both "fill" sequences is equal to the one in the missing genes sequence; only the order in which 1's and 2's appear is different.

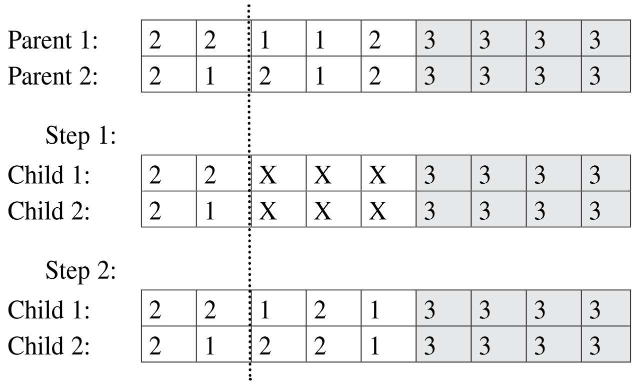
Mutation

After applying crossover, two types of mutation were tested: exchange (EX) and shift (SH) mutation. Mutations are applied to a child with a varying probability p_m (see Reeves, 1995). The mutation rate is calculated as:

$$p_m = 1 - \frac{fit_{best}}{fit_{child}} \qquad (5)$$

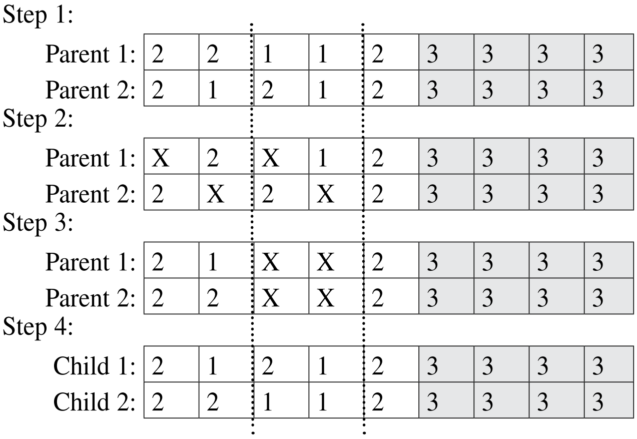
where fit_{best} represents the fitness function value of the chromosome that has yielded the best result and fit_{child} the fitness function value of the child that requires mutation. The chromosome with the fitness function value closer to the fit_{best} would have a lower p_m than the one with the fitness function value closer to the worst value.

The exchange mutation is performed by swapping the places of genes (chosen randomly) to produce a feasible solution (see Figure 7). The shift mutation performs a shift of one gene (chosen randomly) to the right or left for a random number of places (see Figure 8). Again, the mutation can only be performed inside a section.



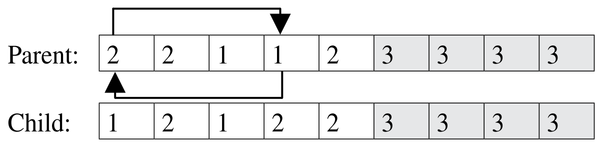
Parent 1:	2	2	1	1	2	3	3	3	3
Parent 2:	2	1	2	1	2	3	3	3	3
Step 1:									
Child 1:	2	2	X	X	X	3	3	3	3
Child 2:	2	1	X	X	X	3	3	3	3
Step 2:									
Child 1:	2	2	1	2	1	3	3	3	3
Child 2:	2	1	2	2	1	3	3	3	3

Figure 5: An example of the CUT crossover in Section 1.



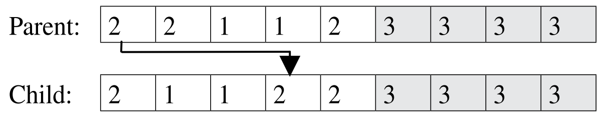
Step 1:									
Parent 1:	2	2	1	1	2	3	3	3	3
Parent 2:	2	1	2	1	2	3	3	3	3
Step 2:									
Parent 1:	X	2	X	1	2	3	3	3	3
Parent 2:	2	X	2	X	2	3	3	3	3
Step 3:									
Parent 1:	2	1	X	X	2	3	3	3	3
Parent 2:	2	2	X	X	2	3	3	3	3
Step 4:									
Child 1:	2	1	2	1	2	3	3	3	3
Child 2:	2	2	1	1	2	3	3	3	3

Figure 6: An example of the LOX in Section 1.



Parent:	2	2	1	1	2	3	3	3	3
Child:	1	2	1	2	2	3	3	3	3

Figure 7: An example of EX mutation in Section 1.



Parent:	2	2	1	1	2	3	3	3	3
Child:	2	1	1	2	2	3	3	3	3

Figure 8: An example of SH mutation in Section 1.

Population size

The effects of setting the parameters of evolutionary algorithms (EA) has been the subject of extensive research by the EA community; recently there has been much attention paid to self-calibrating EAs that can adjust their parameters on-the fly (see e.g., (Eiben et al., 1999) and (Eiben and Smith, 2003) for review). The most attention and most publications have been devoted to the adjustment of parameters of variation operators. Adjusting population size is much less popular, even though there are biological and experimental arguments to suggest that this would be rewarding. Looking at it technically, population size is the most flexible parameter in natural systems: It can be adjusted much more easily than, for instance, the mutation rate. In evolutionary computing, however, population size is traditionally a rigid parameter (Eiben et al., 2004). The new population resizing mechanism used here was introduced by Eiben et al. (2004) and is based on improvements of the best fitness in the population. With fitness improvement, the algorithm becomes more biased towards exploration; increasing the population size, short term lack of improvement makes the population smaller, but stagnation over a longer period causes populations to grow again. Technically, this approach applies three kinds of changes in the population size:

If the best fitness in the population increases, the population size is increased proportionally to the improvement and the number of evaluations left until the maximum allowed. The formula used for calculating the growth rate GR_1 for is:

$$GR_1 = incFact \cdot (maxEvalNum - currEvalNu\ m) \cdot \frac{maxFitness_{new} - maxFitness_{old}}{initMaxFit\ ness} \qquad (6)$$

where *incFact* is an external parameter from the interval (0,1), *maxEvalNum* and *currEvalNum* denote the given maximum number of fitness evaluations and the current evaluation number. *maxFitnessnew*, $maxFitness_{old}$ and *initMaxFitness* are the best fitness values in the current generation, the same in the preceding generation and the best fitness value in the initial population. (Note that we assume the existence of *maxEvalNum*, which is very often present. In case it is not given, a very large number can be used instead.)

The population size is increased by a factor GR_2 if there is no improvement during the last V number of evaluations. In principle, the mechanism to increase the population size in this step can be defined independently from the previous one; in fact, we use the same growth rate, i.e. GR_1.

If neither 1. nor 2. was executed, then the population size is decreased. For the decrease rate (DR), a small percentage of the current population size is used, e.g. (1-5%).

In addition, we added a war or disease process in the population-growth process, similar to the one introduced by Shi et al. (Reeves, 1995). It is known that the natural environment does carry some capacity that cannot be exceeded; hence, we shrunk the population to its initial size after the population exceeded a user-defined maximum population size.

Results

The experiment was performed on a computer with Intel Core2 Quad 2.4GHz processor and 4GB RAM. The GA optimization algorithm was implemented with MS Visual Studio 2008 in C# technology.

Twenty-one were machines needed to perform JSS. Three JSSP configurations were tested during the experiment:

1. BOM1 – 96 jobs divided into four sections with a chromosome length of 447 genes.
2. BOM2 – 101 jobs divided into four sections with a chromosome length of 558 genes.
3. BOM12 – a combination of BOM1 and BOM2.

Most jobs in BOM1 and BOM2 require six operations, while the minimum is one and maximum is 13 operations per job. Jobs with no operations are considered assembly jobs that do not require any operation to be performed but are needed to satisfy the precedence constraint. Approximately 20% of jobs do not require recirculation.

Four different GA configurations were tested; each GA configuration utilizes different GA methods as shown below:

1. No mirroring applied, roulette selection utilized, a population of 50 individuals, CUT and EX methods applied, CUT method applied in 80% of cases.
2. Mirroring applied, roulette selection utilized, a population of 100 individuals, CUT, LOX and EX methods applied. In the crossover process, the CUT method was applied in 60% of cases and LOX method in 40% of cases.
3. Mirroring applied, FRD selection utilized, a population of 100 individuals, CUT, LOX, EX and SH methods applied. In the crossover process, the CUT method was applied in 60% of cases and LOX method in 40% of cases.
4. Mirroring applied, FRD selection utilized, variable population with an initial value of 50 individuals, CUT, LOX, EX and SH methods applied. In the crossover process, the CUT method was applied in 60% of cases and LOX method in 40% of cases.

Each JSSP configuration was run 10 times for each GA configuration, and average makespan and runtime were calculated. Then for each GA configuration, the results of each JSSP configuration were summed (BOM1+BOM2+BOM12) to yield the makespan and runtime for each GA configuration.

The results of the production scheduling simulation are presented in Figure 9, showing makespan progress through the research by utilizing different GA methods. Four different GA configurations are presented and compared, regarding makespan (solid line) and runtime (dashed line).

The first configuration was our starting point. Obviously, it yielded the worst makespan (3995 min) of all GA configurations and the shortest simulation runtime (131s), due to the smaller population size and the utilization of fewer and simpler GA methods. The second configuration, including mirroring of initial population, increase of population to 100 individuals, LOX and EX method, yielded a far better makespan (3489 min) and a longer runtime (165s) due to increased population size and application of more GA methods. The third configuration, reverting from roulette to FRD selection and including the SH method, yielded another significant improvement in makespan (3030 min) while greatly increasing the runtime to 339s, due to the more complex selection method and an application of another GA method. The fourth configuration, reverting from a fixed population size to variable, yielded a slight improvement in makespan (3026 min)

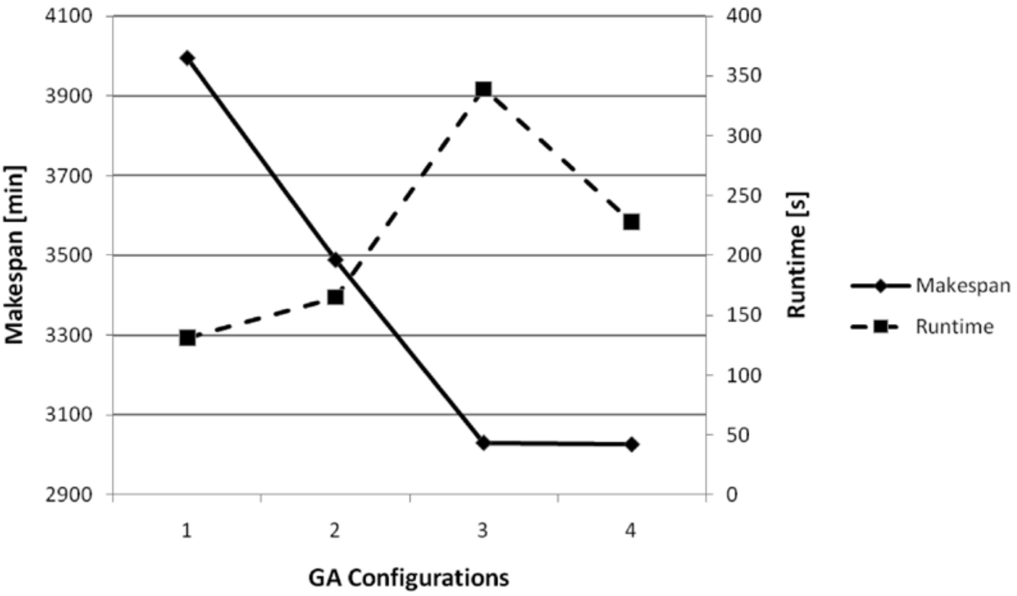


Figure 9: A comparison of makespan and runtime for different GA configurations.

while the runtime (228s) significantly improved over the third configuration.

One can notice a significant improvement in makespan from the first to the fourth configuration, with a decrease of 969 min or 24.3%. In contrast, the runtime increased by 97s or 74%. However, the makespan improvement more than compensates for the runtime increase.

3 Web application development

A web application is an application that is accessed via a web browser over a network such as the internet or an intranet. Web applications are popular due to the ubiquity of web browsers, and the convenience of using a web browser as a client. The ability to update and maintain web applications without distributing and installing software on potentially thousands of client computers is a key reason for their popularity, as is the inherent support for cross-platform compatibility. The drawbacks are the inaccessibility of a web application if the internet connection is broken and vulnerability to threats emerging on the internet. Web applications are based on a client-server principle, in which the client-server characteristic describes the relationship of cooperating programs in an application. The server component provides a function or service to one or many clients, who initiate requests for such services.

Our web solution is based on the web application model shown in Figure 10. The client sends HTTP requests to the Apache web server through the user interface. The Apache server interconnects with MySQL Database to retrieve or store the data, and with a scheduling engine to perform the scheduling process. The processed data is then sent back to the user and displayed on the user interface.

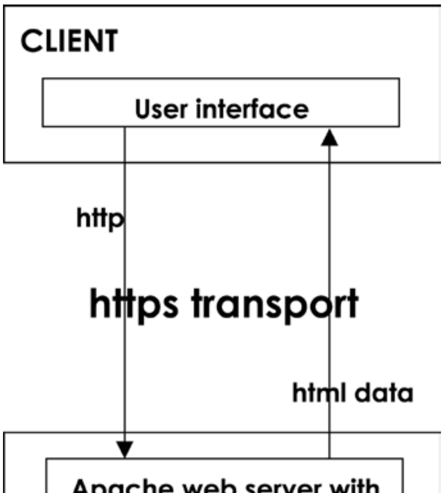


Figure 10: Web application model in our case

Together with a secure web server, reliable database, and an effective scheduling engine, the user interface is a key to application usability. The user interface should include six functional areas: data entry, data display, sequence control, user guidance, data transmission, and data protection (Smith and Mosier, 1984). The system's user interface was developed with the intention of decreasing the cognitive load by using a well-structured presentation of strategic information, which can greatly improve the utilization of such a system (Schuurman et al., 2008). The user interface will be effective if all user actions are performed with minimal effort, if all users are satisfied with it and have a feeling that it offers support in what they do (Desemo et al., 2008). Hence, the user interface is one of the key factors of a successful information system, and web applications are based on a functional user-interface. An example of our user interface is shown in Figure 11.

The division of system functionality into modules enables understanding of a complex system, in which a vast number of variables are hard to control. Complex problems can be solved efficiently if they are divided into controllable parts. This also holds for software and its modules (Solina, 1997). Our web solution consists of seven main modules (Knaflič, 2009):

1. User management (adding new users, granting privileges, etc.),
2. News management (news categories, adding new news, etc.),
3. Resource management (workplaces, operations, jobs, etc.),
4. Customer management (adding new customers, modifying existing ones, etc.),
5. Project management (adding new projects, modifying existing ones, etc.)
6. Planning module (setting planning attributes, performing scheduling, etc.),
7. Reporting module (generating reports on the basis of production planning module).

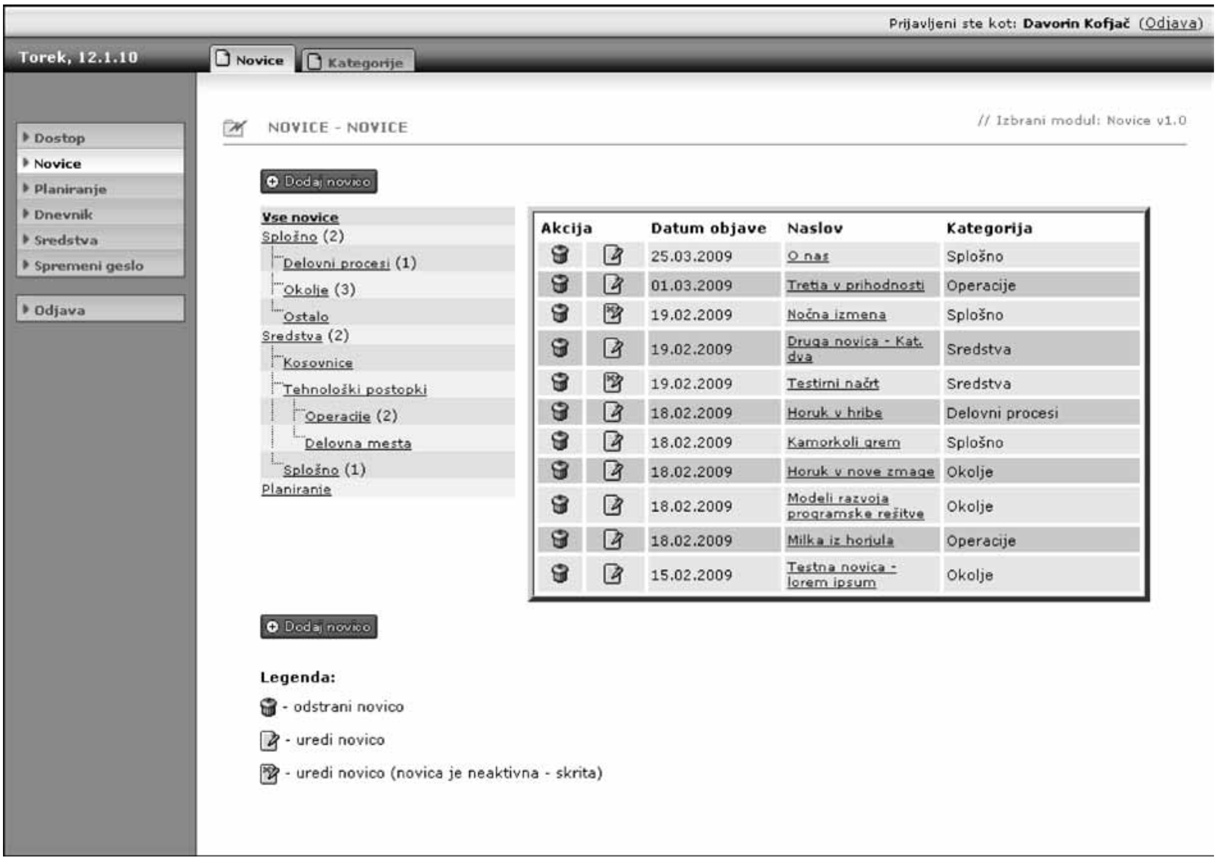
Modules are then grouped into three tiers regarding user privileges (Figure 12):

1. General – visible to all users, providing basic news and company information,
2. Registered users – visible to those whose functionality depends on user privileges, providing detailed information, reporting, and enabling production-related activities with touchscreens in the production hall,
3. Administrative – visible only to administrators, where user, resources, news, customers, and projects management is available, as well as production planning.

With the web application design described above, we have successfully developed a system that covers all the main functionalities that completely support the dynamic production scheduling process.

4 Conclusion

Production scheduling systems are a specific form of decision support system. The market is full of commercial tools that require major investment if they are to be implemented in production systems. Furthermore, such tools usually impose



Prijavljeni ste kot: **Davorin Kofjač** (Odjava)

Torek, 12.1.10

Novice | Kategorije

- Dostop
- Novice
- Planiranje
- Dnevnik
- Sredstva
- Spremeni geslo

- Odjava

NOVICE - NOVICE

// Izbrani modul: Novice v1.0

⊕ Dodaj novico

Vse novice
Splošno (2)
  Delovni procesi (1)
  Okolje (3)
  Ostalo
Sredstva (2)
  Kosovnice
  Tehnološki postopki
    Operacije (2)
    Delovna mesta
  Splošno (1)
Planiranje

Akcija		Datum objave	Naslov	Kategorija
🗑	✎	25.03.2009	O nas	Splošno
🗑	✎	01.03.2009	Tretja v prihodnosti	Operacije
🗑	✐	19.02.2009	Nočna izmena	Splošno
🗑	✎	19.02.2009	Druga novica - Kat. dva	Sredstva
🗑	✐	19.02.2009	Testirni načrt	Sredstva
🗑	✎	18.02.2009	Horuk v hribe	Delovni procesi
🗑	✎	18.02.2009	Kamorkoli grem	Splošno
🗑	✎	18.02.2009	Horuk v nove zmage	Okolje
🗑	✎	18.02.2009	Modeli razvoja programske rešitve	Okolje
🗑	✎	18.02.2009	Milka iz horjula	Operacije
🗑	✎	15.02.2009	Testna novica - lorem ipsum	Okolje

⊕ Dodaj novico

Legenda:

🗑 - odstrani novico

✎ - uredi novico

✐ - uredi novico (novica je neaktivna - skrita)

Figure 11: An example of News management in the administration tier

high costs for support and maintenance. Small and medium enterprises (SMEs) do not have the funds necessary to implement such tools, but have a great need for them, because they must cope with limited inventory capacity, limited workforce and other resources, highly flexible production and so on. Therefore, web solutions that are easy to implement, platform independent, accessible from anywhere, and, most importantly, have low maintenance costs are most appropriate for SMEs.

Our web solution to support dynamic production scheduling was developed with open source technologies, such as PHP and MySQL; only the powerful computational scheduling engine was implemented with a commercial product, Microsoft Visual Studio. The web solution supports resources management, such as users, workforce, jobs etc., production scheduling (supported by genetic algorithms), reporting and real-time activities in the production process, such as marking specific operations as complete on a touchscreen in the production hall. The web solution is built in a modular way, with an intuitive and user-friendly graphical user interface to support man-machine interaction.

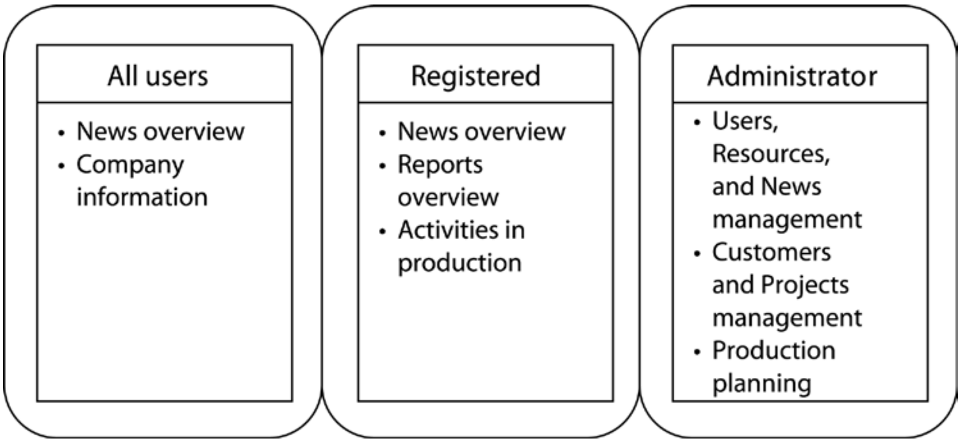


Figure 12: Three tier web application configuration regarding user privileges

In our case, production scheduling is based on genetic algorithms. We have proven that genetic algorithms (GAs) perform well in our case of a make-to-order production process, which can be described as a job-shop process. GAs are very effective in solving job-shop scheduling problems because they are able to explore a vast search space with the goal of finding an optimal/near optimal schedule. We have shown that only by applying different GA methods can one achieve a 25% lower makespan that can be performed in a reasonable amount of time, because real-time scheduling is required in such a dynamic and flexible production system.

Acknowledgement: *This research was supported by the Ministry of Higher Education, Science and Technology of the Republic of Slovenia (Contract No. Z5-0015-0586-08). Our sincere thanks also go to Mr. Bolčič and his team who initiated this research.*

References

Akkermans H. A., Bogerd P., Yucesan E. & van Wassenhove L. N. (2003). The impact of ERP on supply chain management: Exploratory findings from a European Delphi study, *European Journal of Operational Research*, 146(2), 284-301, DOI: 10.1016/S0377-2217(02)00550-7.

Bernik, I. (2001). Večkriterijsko razporejanje z genetskimi algoritmi in vizualno simulacijo s Petri mrežami [Multicriteria production scheduling with genetic algorithms and visual simulation with Petri Nets], Doctoral Dissertation (in Slovenian), University of Maribor, Faculty of Organizational Sciences.

Breskvar U. & Kljajić M. (2001). Možnost uporabe simulacijskega modela v naročniški proizvodnji [Possibility of using a simulation model in a make-to-order production]. *Management in globalizacija : zbornik posvetovanja z mednarodno udeležbo*. Editor: Vukovič, G., Portorož 2001, p. 1075-1079, Kranj: Moderna organizacija.

Buchmeister, B., Palčič, I. & Pavlinjek, J. (2008). Nekonvencionalne metode terminiranja proizvodnih procesov [Unconventional methods for production process termination]. *Orodjarstvo 2008, Organizacija kot gonilo poslovnih izboljšav : dobavitelj - kupec - orodjar* : (proceedings). Editors: Polajnar, A., Bračko, B., Junkar, M., Portorož 7-9 October 2008, p. 67-74, Ljubljana: GZS, Združenje kovinske industrije, Odbor za orodjarstvo; v Mariboru: Fakulteta za strojništvo.

Caseau Y. & Laburthe F. (1995). *Disjunctive scheduling with task intervals*. Technical report, LIES, Ecole Normale Superieure de Paris.

Chen, H., Ihlow, J. & Lehmann, C. (1999). A Genetic Algorithm for Flexible Job-Shop Scheduling, *Proceedings of the 1999 IEEE International Conference on Robotics & Automation*, Detroit, Michigan, 10-15 May 1999, pp. 1120-1125, Los Alamitos, CA: IEEE Computer Society.

Choi H. R., Kim H. S., Park B. J., Park Y. J. & Whinston A. B. (2004). An agent for selecting optimal order set in EC marketplace, *Decision Support Systems*, 36(4), 371 – 383, DOI: 10.1016/S0167-9236(03)00027-7.

Choi S.H. & Yang F.Y. (2006). A filter search algorithm based on machine-order space for job-shop scheduling problems, *Journal of Manufacturing Technology Management*, 17(3), 376-392, DOI: 10.1108/17410380610648326.

Davis L. (1985). Job shop scheduling with genetic algorithms. *Proceedings fo the First Internacional Conference on Genetic Algorithms and their Aplications*. Editor: Grefenstette, J.J., Pittsburgh, PA, 24-26 July 1985, pp. 136-140, Hillsdale, NJ, USA: L. Erlbaum Associates Inc.

Derigs U. & Jenal O. (2005). A GA-based decision support system for professional course scheduling at Ford Service Organisation. *OR Spectrum*, 27(1), 147-162, DOI: 10.1007/s00291-004-0180-8.

Deserno T. M., GÏld M. O., Plodowski B., Spitzer K. & Wein B. B. (2008). Extended Query Refinement for Medical Image Retrieval. *Journal Journal of Digital Imaging*, 21(3), 280-289, DOI: 10.1007/s10278-007-9037-4.

Dorn J., Slany W. (1994). A flow shop with compatibility constraints in a steel making plant. *Intelligent Scheduling*. Editors: Zwebenm M., Fox, M., pp. 629-654, San Francisco: Morgan Kaufmann.

Eiben, A.E., Marchiori, E., & Valko, V.A. (2004). Evolutionary algorithms with on-the-fly population size adjustment. Lecture Notes in Computer Science, Vol. 3242, Parallel Problem Solving from Nature - PPSN VIII, 8th International Conference Proceedings, Editors: Yao, X. et al., Birmingham, 18-22 September 2004, pp. 41-50, Berlin: Springer.

Gen, M. & Cheng, R. (1997). *Genetic Algorithms & Engineering Design*. New York: John Wiley & Sons, Inc.

Jain A. S. & Meeran S. (1998). *A state of the art rewiew of job-shop scheduling techniques*. Dundee, Scotland: Technical report, Department of Applied Physics, Electronic and Mechanical Engineering, University of Dundee.

Kljajić M., Breskvar U. & Bernik I. (2001). Planiranje proizvodnje s simulacijo in genetskimi algoritmi [Production planning with simulation and genetic algorithms]. *Zbornik desete Elektrotehniške in računalniške konference ERK*, Editor: Zajc, B., Portorož, 24-26 September 2001, pp. 311-314, Ljubljana: IEEE Region 8, Slovenska sekcija IEEE.

Knaflič, A. (2009). Razvoj spletne aplikacije za dinamično razporejanje proizvodnje v malih in srednje velikih podjetjih [Development of a web application for dynamic production scheduling in small and medium enterprises], Master's thesis (in Slovenian), University of Maribor, Faculty of Organizational Sciences.

Kofjač, D. & Kljajić, M. (2008). *Application of genetic algorithms and visual simulation in a real-case production optimization. WSEAS Transactions on Systems and Control*, 12(3): 992-1001.

Kramberger T. & Rosi B. (2007). Do Managers have Enough Quality Information for Decision-Making. *Organizacija*, 40(3): 207-217.

Ljubič T. (2000). *Planiranje in vodenje proizvodnje [Production planning and control]*. Kranj: Moderna organizacija.

Marinho J., Braganca A., Ramos C. (1999). Decision support system for dynamic production scheduling. *Proceedings of the 1999 IEEE International Symposium on Assembly and Task Planning*, Editor:, Porto, 21-24 July 1999, pp. 424-429, Evanston: IEEE Robotics & Automation Society.

Na D.-G., Jang W., Kim D.-W. (2003). Development of a resource planning system for compound semiconductor wafer manufacturing. *Journal of Materials Processing Technology*, 138(1-3), 372-378, DOI: 10.1016/S0924-0136(03)00101-8.

Naso D., Surico M., Turchiano B. & Kaymak U. (2007). Genetic algorithms for supply-chain scheduling: A case study in the distribution of ready-mixed concrete, *European Journal of Operational Research*, 177(3), 2069-2099, DOI: 10.1016/j.ejor.2005.12.019.

Reeves C. (1995). A genetic algorithm for flow shop sequencing, *Computers and Operations Research*, 22(1), 5-13.

Smith, S. L. & Mosier, J. N. (1984). *Design Guidelines for User-System Interface Software*. Technical Report ESD-TR-84-190, Hanscom Air Force Base, MA: USAF Electronic Systems Division.

Solina F. (1997). *Projektno vodenje razvoja programske opreme [Project management of software development]*. Ljubljana: Fakulteta za računalništvo in informatiko.

Stawowy A., Duda J., Osyczka A. & Jankowski R. (2007). Web-based Capable to Promise system. *Information technologies in economics and innovative management*. Editor: Dura, J.T., pp. 42-56, Krakow: AGH University of Science and Technology Press.

Schuurman N., Leight M. & Berube M. (2008). A Web-based graphical user interface for evidence-based decision making for health care allocations in rural areas. *International Journal of Health Geographics*, 7(1), 49-61, DOI: 10.1186/1476-072X-7-49.

Shi, X.H., Wan, L.M., Lee, H.P., Yang, X.W., Wang, L.M. & Liang, Y.C. (2003). An improved genetic algorithm with variable population-size and a PSO-GA based hybrid evolutionary algorithm. *Proceedings of the 2003 International Conference on Machine Learning and Cybernetics*, Xian: 2-5 November 2003, 3: 1735-1740, Los Alamitos: IEEE Computer Society Press.

Tarantilis C.D., Kiranoudis C. T. & Theodorakopoulos N. D. (2008). A Web-based ERP system for business services and supply chain management: Application to real-world process scheduling. *European Journal of Operational Research*, 187(3), 1310–1326, DOI: 10.1016/j.ejor.2006.09.015.

Davorin Kofjač gained his Ph.D. from the University of Maribor in the field of information systems management. He is a researcher and an assistant professor at the University of Maribor, Faculty of Organizational Sciences at the Cybernetics and Decision Support Systems Laboratory. His main research interests include modelling and simulation, decision support systems, operational research and artificial intelligence.

Andrej Knaflič achieved his M.Sc. degree in the field of work process management at the University of Maribor, Faculty of Organizational Sciences where he was a young researcher. He was involved in several national research projects. Currently he is working at Mimovrste d.o.o where his main focus is analytics and quality control.

Miroljub Kljajić is a professor at the University of Maribor, Faculty of Organizational Sciences, in the field of System Theory, Decision Theory and Computer Simulation. He completed his B. Sc., M.Sc, and D.Sc. at the University of Ljubljana, Faculty of Electrical Engineering. He has developed a method of quantitative gait evaluation and a simulation system for decision making support in the business systems. He has been the principal investigator of many national and international projects. He has published over 30 articles in world renowned journals.

Razvoj spletne aplikacije za dinamično razporejanje proizvodnje v malih in srednje velikih podjetjih

Prispevek obravnava razvoj spletne aplikacije za potrebe dinamičnega razporejanja proizvodnje v malih in srednje velikih podjetjih. Medtem ko večja podjetja lahko investirajo v ljudi in ustrezno tehnologijo, ki podpira aktivnosti razporejanja, številna mala in srednja podjetja trpijo prav zaradi pomanjkanja podpore pri upravljanju le-teh. Glavni cilj je bil razvoj stroškovno ugodne, učinkovite rešitve za potrebe razporejanja proizvodnje v malih in srednjih podjetjih s poudarkom na dostopnosti, neodvisnosti od platform in enostavnosti uporabe. Zaradi tega smo se odločili za razvoj spletne rešitve, pri kateri je bil glavni poudarek na razvoju inteligentnega in dinamičnega uporabniškega vmesnika. Rešitev je zgrajena modularno in omogoča dinamično razporejanje naročil v proizvodnjo na podlagi metode umetne inteligence - genetskih algoritmov. Rešitev je razvita kot samostojen informacijski sistem, ki preko administracijskega vmesnika omogoča upravljanje vseh aktivnosti razporejanja. Tako lahko posameznim uporabniškim skupinam določamo ustrezne dostopne pravice in s tem uporabnikom omogočamo izvajanje posameznih aktivnosti. Poleg tega je v nalogi zajetih še pet glavnih funkcionalnosti, ki podpirajo dejavnosti razporejanja. Tako rešitev omogoča popisovanje sredstev s katerimi podjetje razpolaga, uporabo le-teh v procesu planiranja proizvodnje, zbiranje podatkov o proizvodnih aktivnostih, ažurno razpečevanje informacij in izvajanje nadzora nad dogajanjem v sistemu.

Ključne besede: dinamično razporejanje proizvodnje, genetski algoritmi, razvoj spletne rešitve, spletni uporabniški vmesnik, mala in srednje velika podjetja