Hindawi
Mathematical Problems in Engineering
Volume 2018, Article ID 9838560, 14 pages
https://doi.org/10.1155/2018/9838560

Research Article

Innovative Approach in Modeling Business Processes with a Focus on Improving the Allocation of Human Resources

Almir Djedovic, Almir Karabegovic, Zikrija Avdagic, and Samir Omanovic

Faculty of Electrical Engineering, University of Sarajevo, Sarajevo 71000, Bosnia and Herzegovina

Correspondence should be addressed to Almir Djedovic; almir.djedovic@infostudio.ba

Received 4 May 2018; Revised 2 July 2018; Accepted 30 August 2018; Published 19 September 2018

Academic Editor: Ricardo Soto

Organizations can improve efficiency of process execution through a correct resource allocation, as well as increase income, improve client satisfaction, and so on. This work presents a novel approach for solving problems of resource allocation in business processes which combines process mining, statistical techniques, and metaheuristic algorithms for optimization. In order to get more reliable results of the simulation, in this paper, we use process mining analysis and statistical techniques for building a simulation model. For finding optimal human resource allocation in business processes, we use the improved differential evolution algorithm with population adaptation. Because of the use of a stochastic simulation model, noise appears in the output of the model. The differential evolution algorithm is modified in order to include uncertainty in the fitness function. In the end, validation of the model was done on three different data sets in order to demonstrate the generality of the approach, and the comparison with the standard approach from the literature was done. The results have shown that this novel approach gives solutions which are better than the existing model from literature.

1. Introduction

Business Process Management (BPM) is defined as a discipline "supporting business processes using methods, techniques, and software to design, enact, control, and analyze operational processes involving humans, organizations, applications, documents, and other sources of information" [1]. Business process simulation (BPS) plays a significant role in the continuous improvement approach to business process management and it is used in many areas [2].

The results obtained by the simulation depend on the quality of the simulation model and input data accuracy. If inaccurate models are used or poor data are provided as input, decisions based on simulation results may impair operations instead of providing the expected improvements [3]. Some companies do not have well-structured process models and in many cases processes which take place in reality differ from processes presented on paper. The traditional method for getting the workflow of a process is through surveys, interviews, questionnaires, workshops, etc., with the owners of the process and/or participants in the process. This method is time-consuming and costly, it is subjective and error prone. Because of that reason, the process models acquired this way can be differentiated in regard to the real process model. In order to avoid these problems, the paper will use process discovery algorithms for getting the workflow of a process. In order to get the remaining information such as time needed for generating process instances and time needed for the execution of tasks on the activities, statistical analysis methods will be used. Using these methods, statistical distributions will be acquired, as well as distribution parameters which can be used to describe the execution of tasks in the process. This set of information is sufficient for getting the simulation model and for conducting what-if analysis. The focus of this paper is on optimization of the process by allocating the available human resources in the process.

2. Related Work

There are a lot of papers that use metaheuristic algorithms for solving Stochastic Combinatorial Optimization Problems (SCOPs). The main characteristic of these problems is that a part of the information about data is unknown and that it is presented in a form of a probability distribution function.

Metaheuristic algorithms like Simulated Annealing (SA), Ant Colony Optimization (ACO), Evolutionary Computation (EC), etc. have replaced approaches based on mathematical and dynamic programing for solving SCOPs. In these problems, uncertainty can be present in different ways and it is included in real problems so they can be described in a more precise way. Uncertainty can be present in the fitness function and/or in constraints. In [4] Guimei and Haijun solved the SSRA (stochastic resource allocation problem) problem in complex systems. The authors used the algorithm of differential evolution combined with local search in order to create a hybrid optimizer. The algorithm proved itself to be successful on the tested scenarios. In [5] the authors solved the problem multicriteria budget allocation under uncertainty. The authors took into consideration uncertainty in models using robust weighted sum approach. For solving the model, the authors used a sample average approximation approach together with a cutting surface method. In [6], authors solved class of revenue management problems in systems with reusable resources. The authors suggested 2-approximation algorithm for revenue management models in which resources can be reused, and [7] was expanded so advance reservations are possible. Some authors use approximation algorithms for solving this problem. Dye et al. in [8] considered the service-provision problem. In this paper, uncertainty was presented in requests for services. The authors used two-approximation algorithms for solving this problem. In papers noise is most frequently described with a normal distribution. Central Limit Theorem can be exploited to create a normal distribution regardless of the underlying noise distribution.

In this paper, Differential Evolution (DE) algorithm was used to solve the problem. After that, the used algorithm was adjusted to take uncertainty in the fitness function into account. Uncertainty in the fitness occurs because a stochastic simulation model described with probability functions of distribution is used. The algorithm is modified so it takes into account noise which occurs in the fitness function. A detailed description of the expanded algorithm is given in Section 3.

Resource Allocation Problem (RAP) can be defined as follows: N units of resources need to be allocated on n activities. By allocating resources on each activity costs arise, which are the function of allocated resources. It is necessary to find the optimal resource allocation in order to minimize the total cost. For decreasing convex cost function, the problem can be solved in polynomial time by a simple greedy algorithm in $O(n^2)$ [9]. Therefore, it is not possible to optimally solve an instance of the problem within the very restrictive time limit and the problem belongs to the category of the NP-hard problem. Also, RAP is NP-hard since the classical 0–1 knapsack problem can be understood as a special case of RAP where all-time intervals are identical. The mathematical proof that this problem belongs to the category of NP-hard can be found in [9].

Managing human resources in the organization has become one of the main methods of business process improvement. Practical application of the solution of the human resource allocation problem (HRAP) can be found in different real areas: project management, production systems, hospitals, maintenance systems, educational institutions, and so on [10]. This problem can be viewed as a subset of a larger problem known as a resource allocation problem (RAP) which takes into account material as well as nonmaterial resources. In this paper, the focus will be on the allocation of human resources in business processes. For solving this problem, the authors use exact methods, heuristic algorithms, and metaheuristic algorithms or combine many different methods. Because HARP falls into the class of NP-hard combinatorial problems, exact methods [11, 12] are often unable to find the solution. Metaheuristic algorithms are most commonly used for solving this problem. Gunawan & Ng [13] used simulated annealing (SA) and tabu search (TS) algorithms for solving the teacher assignment problem. The algorithm's execution consisted of two phases. In the first phase, the teachers are allocated to the courses and the number of courses which is supposed to be assigned to each teacher was also determined. In the second phase, the teachers were allocated to the course sections in order to balance the load of the teachers. Authors in [14] used genetic algorithm (GA) to solve resource allocation problem. The genetic algorithm managed to find the optimal solution, but the number of iterations and the execution time of the algorithm for finding the optimal solution drastically increased with the increase of the problem complexity. Park et al. [15, 16] aspired to solve the human resource allocation problem in software project development, including practical constraints, using genetic algorithm (GA). Diakoulakis, Koulouriotis & Emiris [17] presented evolution strategies for solving the constrained resource project scheduling problem. A practical approach for solving real complex constrained resource project scheduling problems is presented by Pantouvakis & Manoliadis [18]. The authors have developed a heuristic method based on CPM (Critical Path Scheduling) and leveling algorithms. An additional review of the papers which use metaheuristics in project and construction management is given in the work of Liao, Egbelu, Sarker, & Leu [19]. The authors concluded that the majority of the assumptions which are used in the models are far from reality, as well as that the used text examples are small compared to real-world problems. The main difference between this paper and the others is that in our case the fitness function for optimization contains noise. Noise occurs because of the use of a stochastic simulation model, so it is necessary to take that into consideration in the use of metaheuristic algorithms.

3. Business Process Improvement through Human Resource Allocation

For improving business processes through resource allocation the model shown on Figure 1 will be used. Data about the execution of business processes can be in different formats: database, textual documents and so on. Data can contain errors (noise in the data), so it is necessary to clear the data from any errors. Also, in some cases events are not grouped into process instances, so it is necessary to do data preprocessing. It is necessary to format the data into an appropriate MXML (Mining Extensible Markup Language)

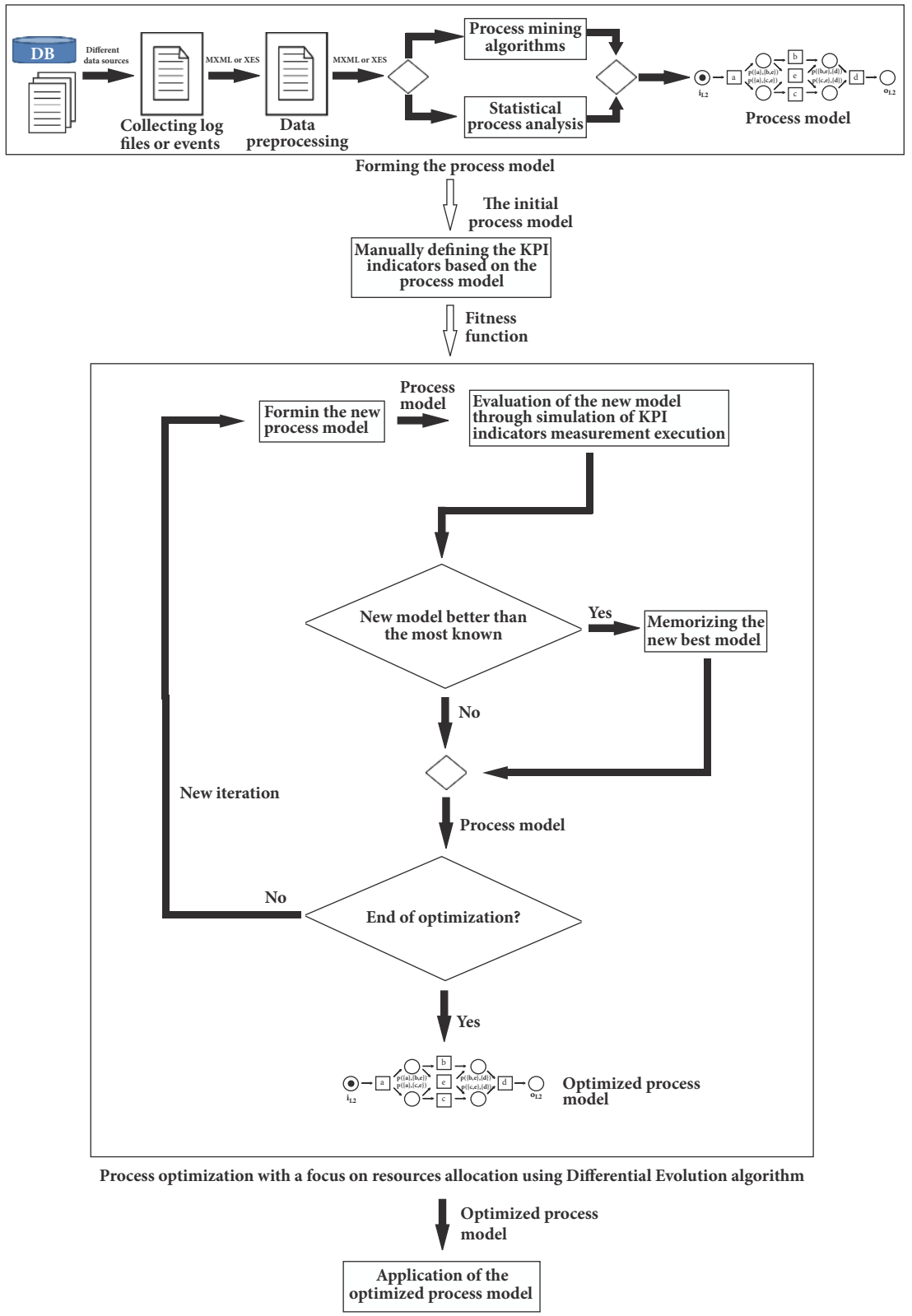

Figure 1: Improving business processes using resource allocation.

or XES (Extensible Event Stream) format in order to use process discovery algorithms.

In this paper, data about the execution of business processes is saved in a database based on the execution of real processes in organizations, where they are formatted into log files in MXML format. Log files are placed on 4TU.ResearchData in order to be available to the rest of the researchers.

For identification of the current process model, the way it is executed in reality, techniques of process mining, and statistical analysis are used. Using process mining, it is possible to get the process flow from the data, and the information for developing a stochastic process model is acquired using statistical analysis. Application of the listed techniques results in a process model the way it is executed in reality. After identification of the current process model, the criteria of optimization are manually defined (KPI, Key Performance Indicators), which will be used for forming the fitness function. In this step, constraints will be defined related to the optimization of the business process. The previously acquired process model along with a defined fitness function and constraints present an entrance to optimization of the business process using resource allocation. A new process model with different resource allocation is formed in each new step of the process optimization. It is checked whether the process model meets the constraints and it is evaluated in the step of evaluation of the new model through simulation of KPI indicator measurement execution. If the new process model is better than the currently best known, then it is memorized and it becomes the best known model. This process is continued until the requirement for the end of the optimization is met. The requirements for the end of the optimization can differ: number of iterations, time of algorithm execution, stagnation of the best result, and so on. After the optimization is finished, the best solution is chosen and it is ready for application. Using the stochastic simulation model causes the occurrence of noise in the fitness function. Noise occurs because the process model is described with a probability distribution function, so the output of the process model can differ for the same input parameters of the model. Because of that, all the optimization algorithms will be adjusted so that they can take into account uncertainties which occur in the fitness function. What follows is the detailed description of the listed steps.

4. Building a Simulation Model

The first step towards improving business processes by resource allocation is the building of a simulation model. A simulation model should manage to replicate a real-world scenario of the process as accurately as possible. Simulation parameters are process workflow, arrival rate, task execution time, number of resources involved in the process, and their cost. Different tools for BPS provide a different set of simulation parameters. In this paper, the previously mentioned simulation parameters will be observed.

4.1. Discovering Process Workflow. Not all companies today have documented business processes. Also, in companies which have documented processes, the execution of processes in reality can be different from the processes from the documentation. The reasons for that could be changes of policy, processes which are not updated in the documentation and so on. In this paper process discovery algorithms will be used for acquiring process models like they are executed in reality. Process mining is a technique for extracting a process model based on its real time behavior recorded in the event log. Event logs contain the following information: process instance ID, workflow model element, EventType, Originator. The process instance ID represents a unique identifier of the process instance, and the workflow model element represents the name of the activity in the process. EventType represents the type of the event where two values are possible: start (when the event started) and complete (when the event finished). Timestamp represents the time of the event, while the attribute Originator represents the user in the process.

The two standard formats for event logs are MXML and XES. Process discovery algorithms acquire workflow processes from event logs. There are several process discovery algorithms: α mining [20], genetic mining [21], and heuristic mining [22]. The heuristic miner presents an upgraded α algorithm, taking into consideration the frequency of activity sequences. The heuristic miner is resistant to errors in log events. Using the heuristic miner the main process flow can be acquired, and exceptions are not taken into consideration and are solved by specifying certain parameters in the algorithm.

4.2. Finding the Simulation Parameters. After the real process model is acquired, it is necessary to get the other simulation parameters. The process of getting the simulation parameters is described in [23]. Generally, events can contain different attributes. In this case, a minimal number of attributes sufficient for the analysis were observed. The first step is to extract the information about the time between the generation of process instances and the time of the duration of tasks in the process, for which the following rules are used (Table 1).

After extracting the information about time, it is necessary to define the probability distribution functions. Defining these functions consists of the following three steps.

(a) Choosing the Function. One of the methods for determining the probability distribution function is the Cullen and Frey graph [24]. This method is based on high-order moments: the third (skewness) and fourth (kurtosis).

Nonzero value of the skewness parameter indicates the existence of symmetry in the probability distribution function. On the other hand, the kurtosis parameters indicates the existence of a tail in the function of distribution, which is compared to the tail of the normal distribution which is 3. Cullen and Frey graph represents skewness and kurtosis parameters, and, based on their relation, it is possible to determine the functions of distribution.

Also, one of the methods are histograms. Histograms can be used in determining whether data is symmetric, right skewed, or left skewed and then to choose the distribution

Table 1: Rules for extracting time information.

Rule	Description
Rule 1	IF w_{ij}=*Register* AND $w_{(i+1)j}$=*Register* THEN $x_i = t_{(i+1)j} - t_{ij}$ END IF
Rule 2	IF $w_{ij} = w_{i(j+1)}$ AND w_{ij}=start AND $w_{i(j+1)}$=complete AND $o_{ij} = o_{i(j+1)}$ THEN $x_i = t_{i(j+1)} - t_{ij}$ END IF

function. Q-Q and P-P graphs can also be used along with these methods. Normally in this step several potential functions of distribution are chosen, and the selection between them is done during the final step using statistical tests.

(b) Determining the Parameters of the Distribution. For distributions chosen under (a), it is necessary to determine the parameters which describe them. There are different methods used for estimating the parameters: maximum likelihood estimator (MLE), Bayesian estimator, Minimum Mean-Square Error (MMSE), and method of moments (MOME). This paper uses three methods for parameter estimation: MLE, MOME, and maximum goodness-of-fit (MGE).

(c) Goodness-of-Fit Test. After several potential functions of distributions have been chosen in the first step and their parameters have been determined, this step uses statistical tests in order to determine the distribution which best describes the random variable. These tests work based on the comparison between the theoretical and estimated function of distribution. The distribution which is most similar to the estimated distribution function is chosen. This paper will use the three most popular tests: Kolmogorov Smirnov (KS), Carmén von Mises (CvM), and Anderson Darling (AD). CvM and AD are used when it is necessary to choose a function of distribution which best describes matching in the tails of the distributions and KS for better matching in the centers of the distributions.

5. Improving Business Processes with Allocation of Human Resources

There are many different ways business processes can be improved. In this paper, the improvement of the process is accomplished by human resource allocation. The idea of this improvement is to identify bottlenecks in the process and then improve the process by reallocating of the existing resources or by adding new ones in terms of the defined key performance indicators (KPI). The following contains the description of the problem.

5.1. Problem Description. Business processes are quantitatively and qualitatively measured by KPI indicators. Using KPI indicators, organizations measure the accomplished strategic goals. In general, suppose that the fitness function is a combination of two or more KPIs of interests. The problem of finding the minimum of the fitness function f is observed; if it is necessary to find the maximum value, it is sufficient to observe the fitness function f. Suppose that the process contains n number of activities and that x_i is the number of human resources on the i-th activity in the process. On every activity in the process, there is a constraint in terms of maximal and minimal number of resources. This constraint depends on the work place and on the organization whose business process is observed etc. This can be written in the following form:

$$\alpha_i \le x_i \le \beta_i, \quad \alpha_i, \beta_i \in \mathbb{N} \ \forall i = 1, \ldots, n \tag{1}$$

where α_i is the minimal number of resources and β_i is the maximal number of resources on the i-th activity in the process, respectively. In this paper, the waiting time of the individual clients in the waiting queue will be observed. If w_i represents the waiting time on the i-th activity, then the total waiting time can be represented as sum of all waiting times.

All the waiting times in the process do not necessarily have to be of the same importance. For instance, the waiting time in the queue for the bank counter influences the user's satisfaction more, than when the user is expecting a call from the bank telling him his request has been processed. That is why weight coefficients k_i are introduced. In case that the delay is more significant it is necessary to choose a larger value for the coefficients. The waiting time depends on the number of human resources $w_i = w_i(x_i)$. On the other hand, by engaging additional resources in the process the cost of the process rises. Suppose that the unit cost of the resources on the i-th activity in the process c_i, then the fitness function is equal to the following:

$$f = a \cdot \sum_{i=1}^{n} k_i w_i + \sum_{i=1}^{n} x_i c_i \tag{2}$$

Multiplier a is introduced because in one fitness function the cost of the resources and the waiting time in the queues are combined. The multiplier can be approximated from the delay value which will cause the user's loss and create costs in the organization by losing the client. Therefore, it is necessary to minimize the function f with constraints given with the following:

$$\min_{x_i} \left\{ a \cdot \sum_{i=1}^{n} k_i w_i(x_i) + \sum_{i=1}^{n} x_i c_i \right\}, \quad \alpha_i \le x_i \le \beta_i, \ \alpha_i, \beta_i \in \mathbb{N} \ \forall i = 1, \ldots, n \tag{3}$$

The fitness function is a nonlinear function, because $w_i(x_i)$ is a nonlinear function. The waiting time $w_i(x_i)$ depends on the waiting time of the previous activities, location of the bottleneck in the process, and so on. It is also necessary to keep in mind that x_i takes integers as its values. It is very time-consuming and nearly impossible to find the best distribution of human resources by manually changing the resource allocation, especially if the processes in question are complex and a large number of resources are available. Also, because the final form of the function $w_i(x_i)$ is not known and the fitness function is not smooth, methods such as gradient and derivation cannot be used for finding the minimum of the fitness function. In order to solve this problem, this paper will use a differential evolution algorithm.

The problem is that the fitness function described in this way is deterministic and noiseless. Because the simulation model is stochastic, i.e., described with probability distribution functions, the model output value can differ (in most cases it will differ) for the same input parameters of the model. This occurrence can be described as noise and it is necessary to take it into consideration with problem solving in the fitness function. The noisy fitness function can cause superior candidates to be wrongly believed to be inferior and be eliminated in the selection process. On the other hand, inferior candidates can be believed to be superior and survive into the next generation and reproduce new solutions. So if, for two solutions X_1 and X_2, the results of the simulation show that solution X_1 is better, i.e., that the performances are such that the following relation is valid: $\widehat{f}(X_1) < \widehat{f}(X_2)$ that does not guarantee that it is really true that $f(X_1) < f(X_2)$. Because of that, it is necessary to take statistical significance into account. If function δ represents noise, then the (3) transforms into the following:

$$\min_{x_i} \left\{ a \cdot \sum_{i=1}^{n} k_i w_i(x_i) + \sum_{i=1}^{n} x_i c_i + \delta \right\}, \quad \alpha_i \le x_i \le \beta_i,\ \alpha_i, \beta_i \in \mathbb{N}\ \forall i = 1, \ldots, n \tag{4}$$

The simplest way of solving this problem is resampling and averaging of several samples of the fitness function. This way, the standard error in the fitness function will be reduced. Resampling influences the performances of the algorithm's execution. The larger the number of samples is, the more accurate the value of the fitness function will be calculated, but the time of the algorithm's execution will be bigger as well. In order to solve this problem, this paper will use the *Standard Error Dynamic Resampling* (SEDR) technique, which is described in [25]. In the SEDR algorithm, the number of samples is dynamically assigned for each solution specifically, based on the variance of the fitness function. The algorithm is executed sequentially, after each execution, it is checked whether the stopping condition $se_f < se_{thr}$ is met. If the condition is met that the standard error of the mean is smaller than the threshold, then the algorithm terminates; otherwise additional k samples are assigned and the procedure is repeated. The number of samples m goes from $10, 20, 30, \ldots$, to 150, and the value of the threshold, which is also dynamic, is calculated as follows:

$$se_{thr} = \frac{20000}{\sqrt{m}} \tag{5}$$

The pseudocode of the SEDR algorithm is given in [26]. The fitness function of individuals in the population will be calculated according to the SEDR algorithm. This way of calculating the fitness function leads to error reduction in the value of the fitness function and that will lead to a decrease of the probability of the inferior individual to survive into the next generation and the superior individual to be eliminated. This way of sampling is chosen in order to accelerate the execution of the algorithm, and this way, unlike the standard way of sampling with a constant number samples, works faster because it starts from a minimal number of samples and increases them if necessary. Introducing sampling solves the problem of noise on one hand, but, on the other, it increases the execution time of the algorithm.

5.2. Differential Evolution. The differential evolution algorithm is population based and evolutionary optimization technique. In the DE algorithm, the population consists of NP units $x_{i,G}$, $i = 1, 2, \ldots, NP$ and G is the number of the generation. The first generation is initialized randomly and further generations are acquired using mutation and crossover operations.

Chromosome Encoding and Initial Population. Each chromosome represents a single possible resource allocation in the business process that is being analyzed. The length of the chromosome is equal to the number of activities in the process. The number of resources on each activity is supposed to satisfy the constraints given with (1). The initial population is randomly generated using a uniform distribution in the interval $[\alpha_i, \beta_i]$.

Mutation. There are different versions of mutation and unit recombinations. They are represented using the following notation:

$$DE/x/y/z \tag{6}$$

where x represents the mutated unit, y represents the number of difference vectors, and z represents the crossover method. One of the commonly used mutation operators is $DE/rand/1/bin$:

$$v_{i,G+1} = x_{r1,G} + F \cdot \left(x_{r2,G} - x_{r3,G}\right), \quad i \neq r1 \neq r2 \neq r3 \tag{7}$$

where $r1, r2, r3$ are random numbers from the interval $\{1, 2, \ldots, NP\}$ and F is an amplification factor in interval $[0, 2]$. *rand* means that the vector $x_{r1,G}$ is chosen randomly from the current population. One vector difference $(x_{r2,G} - x_{r3,G})$ and the binomial crossover scheme are used. Two vector differences can be used instead of one, and instead of random values the best unit can be used. In this paper, different strategies for the DE algorithm will be tested.

Crossover. Using the crossover operator trial vectors can be acquired $u_{i,G+1} = (u_{1i,G+1}, u_{2i,G+1}, \ldots, u_{Di,G+1})$, which

come from the target vectors $x_{i,G}$ and mutant vectors $v_{i,G+1}$. Binomial crossover is used for the crossover operation:

$$u_{ji,G+1} = \begin{cases} v_{ji,G+1}, & if\ (r(j) \le CR\ or\ j = rn(i)) \\ x_{ji,G}, & if\ (r(j) > CR\ and\ j \ne rn(i)) \end{cases} \quad (8)$$

where $r(j)$ is the random number from the interval $(0, 1)$, for $j = 1, 2, \ldots, D$, and CR is the probability of the crossover operation.

Selection. Using the selection operator, it is determined whether the trail vectors or target vectors will survive into the next generation:

$$x_{i,G+1} = \begin{cases} u_{i,G+1}, & if\ f(u_{i,G+1}) \le f(x_{i,G}) \\ x_{i,G}, & otherwise \end{cases} \quad (9)$$

In the next generation units have either the equal or better value of the fitness function, so the selection is elitism.

6. Improved DE Algorithm (IDE)

In the differential evolution algorithm, there are 3 control parameters. Those parameters are the crossing probability of CR, amplification factor F, and the size of the population NP. The DE algorithm is very sensitive to the change of these parameters. This paper uses the population adaptation mechanism.

In this paper, the entire population is not updated, but rather half of the population in order to prevent the algorithm from stagnating in the local optimum and to keep the individuals different. Also, a different way of updating the CR operator was defined in accordance with the distance in the population, as well as another way of mutation in individuals.

6.1. Population Adaptation. For measuring the distance between individuals in the population this paper will use Euclidean distance. Euclidean distance is calculated in each generation, according to the following:

$$d(G) = \sum_{i=1}^{NP} \sum_{j=1}^{i-1} \sqrt{\sum_{l=1}^{D} \left(x_{i,l,G} - x_{j,l,G}\right)^2} \quad (10)$$

At the beginning the distance has a large value, because the solutions are randomly distributed in the entire space of the solution. In the process of evolution, the distance decreases from generation to generation, because all the individuals converge towards one solution. When the distance becomes constant that means that either the best solution was found or a stagnation occurred (for example, the algorithm stagnates in the local optimum). If the distance does not change through generations above a certain threshold th_G, then it is necessary to do the population update. Half of the population is updated with a certain probability p_u. The algorithm is tested when a different population size is updated (the whole population, third of the population, and so on), but the performances of the algorithm were best for updating half of the population. The individuals are updated on the current best found, taking into account restrictions:

$$x_{i,G+1} = \begin{cases} x_{best}, & d(G) > th_{NP},\ i \le \dfrac{NP}{2}\ r_i < p_u \\ x_{i,G+1}, & otherwise \end{cases} \quad (11)$$

where r_i is a random number with a uniform distribution in the interval (0,1).

6.2. Evolving Crossover Probability. Apart from the adaptation of the population, changes in the crossover were introduced as well. Crossover probability begins with the initial value and it is evolving through generations. Crossover probability is updated in accordance with the distance between the individuals. When the distance falls below a certain threshold, the crossover probability is updated for a certain step. The threshold changes dynamically according to how many times the crossover probability update happened:

$$CR(G+1) = \begin{cases} CR(G) + 0.1, & d(G) < \dfrac{d_0}{10^{th_{CR}(G)}} \\ CR(G), & otherwise \end{cases} \quad (12)$$

where d_0 is the constant which represents the initial distance between the individuals, and th_{CR} is updated to 1, starting from 0, each time the crossover probability is updated according to the following:

$$th_{CR}(G+1) = \begin{cases} th_{CR}(G) + 1, & CR(G+1) \ne CR(G) \\ th_{CR}(G), & otherwise \end{cases} \quad (13)$$

6.3. Mutation. In the process of mutation, several individuals are generated instead of one, and the individual that survives is the one that has the best value of the fitness function:

$$\begin{aligned} v^1_{i,G+1} &= x_{r1,G} + F \cdot (x_{r2,G} - x_{r3,G}) \\ v^2_{i,G+1} &= x_{r1,G} + F \cdot (x_{r2,G} - x_{r3,G} + x_{r4,G} - x_{r5,G}) \\ v^3_{i,G+1} &= x_{best,G} + F \cdot (x_{r1,G} - x_{r2,G}) \\ v^4_{i,G+1} &= x_{best,G} + F \cdot (x_{r1,G} - x_{r2,G} + x_{r3,G} - x_{r4,G}) \\ v^5_{i,G+1} &= x_{r1,G} + F \cdot (x_{best,G} - x_{r2,G} + x_{r3,G} - x_{r4,G}) \end{aligned} \quad (14)$$

where $r1, r2, r3$, and $r4$ are random numbers.

If the minimum of the function is wanted, then the new individual is chosen according to the following:

$$v_{i,G+1} = \min\left(v^1_{i,G+1}, v^2_{i,G+1}, v^3_{i,G+1}, v^4_{i,G+1}, v^5_{i,G+1}\right) \quad (15)$$

6.4. Experimental Testing Improved DE Algorithm. In order to compare the performances of the proposed DE algorithm (IDE) with the original DE algorithm, a set of benchmark functions with bounds was used. Unimodal as well as multimodal functions were present. A set of tested benchmark

Table 2: Results of the tested algorithm for D=30.

f	DE-1	DE-2	DE-3	DE-4	DE-5	IDE
f_1	0.117 ± 0.016	0.414 ± 0.063	0.076 ± 0.137	0.100 ± 0.019	0.292 ± 0.049	**0.016 ± 0.003**
f_2	88.951 ± 8.369	120.325 ± 9.909	**23.029 ± 7.676**	103.959 ± 9.633	117.572 ± 9.276	34.188 ± 15.263
f_3	0.766 ± 0.206	0.869 ± 0.145	0.097 ± 0.181	0.046 ± 0.063	0.123 ± 0.204	**0.003 ± 0.004**
f_4	0.586 ± 0.152	18.361 ± 3.650	0.908 ± 0.679	1.154 ± 3.573	15.708 ± 5.288	**0.046 ± 0.007**
f_5	63.901 ± 44.318	660.672 ± 231.831	117.854 ± 244.698	50.412 ± 46.496	382.029 ± 142.752	**48.999 ± 31.744**
f_6	31694.276 ± 4364.978	41940.207 ± 6486.947	**2698.609 ± 1915.954**	22355.210 ± 3987.622	39687.656 ± 4442.942	3342.918 ± 1635.032
f_7	0.102 ± 0.014	0.252 ± 0.041	0.049 ± 0.019	0.088 ± 0.017	0.216 ± 0.037	**0.014 ± 0.003**
f_8	0.015 ± 0.005	0.037 ± 0.009	0.010 ± 0.004	0.015 ± 0.006	0.034 ± 0.009	**0.002 ± 0.001**
f_9	0.017 ± 0.004	0.024 ± 0.008	0.008 ± 0.003	0.015 ± 0.005	0.016 ± 0.009	**0.003 ± 0.001**
f_{10}	0.144 ± 0.021	6.804 ± 1.621	0.068 ± 0.019	0.417 ± 0.485	4.966 ± 1.519	**0.016 ± 0.003**

functions with bounds are as follows: Sphere (f_1), Rastrigin function (f_2), Griewank function (f_3), Ackley function (f_4), Rosenbrock function (f_5), Schwefel function No.1.2 (f_6), Brown function (f_7), Csendes function (f_8), AMGN (f_9), and Alpine function No.0.1. (f_{10}). Since the fitness function in the optimization process has noise, in order to simulate the effects of noise and examine how the algorithm behaves in the presence of noise, Gaussian noise with a mean value of 0 and standard deviation of 1 was added to the functions from Table 2, so that the functions have the following form: $f_i(X) + |N(0,1)|, \forall i = 1, \ldots, 10$. The improved algorithm is compared with standard algorithms and the acquired results are shown in Table 2. The algorithms are compared for different dimensions of the problem. The parameters of the improved algorithm were the number of individuals in the population $NP = 30$, maximum number of iterations $MaxIt = 1000$, crossover probability $pCR = 0.3$ and amplification factor F was uniformly distributed in the interval 0.2 and 0.8, the probability of update $p_u = 0.3$, constants $d_0 = 10$, $th_{NP} = 100$ and $th_{CR} = 1$. Parameters of the standard algorithm were the same $NP = 30$, maximum number of iterations $MaxIt = 1000$, and crossover probability $pCR = 0.3$, and amplification factor F was uniformly distributed in intervals 0.2 and 0.8.

The algorithms were run independently 30 times and mean values and standard deviations were calculated. By comparing the results from the table, it can be noted that the proposed algorithm has better performances than the standard DE algorithm. In most cases the IDE algorithm managed to find a better solution than other versions of the standard DE algorithm. In the case where the dimension of the functions was $D = 30$, the IDE algorithm proved best in 8 out of 10 tested cases. The only case where the DE-3 algorithm managed to find a better solution was for functions f_2 and f_6. The algorithm was also tested in the case in which $D = 100$, the algorithm managed to find a better solution than the standard DE algorithm in 7 out of 10 tested cases. The conclusion is that the presented algorithm based on population update and crossover probability update depending on distance between individuals has better performances than the standard DE algorithm.

7. Experimental Evaluation and Discussion of Results

The used dataset contains information about a credit requirement process from a bank in Bosnia and Herzegovina, whose name has been left out due to privacy reasons. This process (dataset) has been selected because it represents one of the main (core) processes in banks. The dataset consisted of information which in its initial form was in tables in the database. Only the data which were necessary for the analysis in this work was extracted from the given dataset: information about the process instance, the name of the event, time of the event, type of the event, and masked data about the participants in the process. First a Java program for converting the dataset to the MXML format is implemented. Also, the data can be converted to this format using the ProM Import Framework. The dataset contains over 150000 events and the data was collected over the period of six months. For determining the statistical functions of the distribution in the process the R programming language was used, and for the implementation of the DE algorithm Matlab 2015a was used.

The first step is acquiring a process model from the data. The Heuristic miner was used for getting the process model, because it is resistant to errors, can work with large log files, and can be used for real log files. The acquired process model is presented in Figure 2, using BPMN (Business Process Modeling Notation). The process begins by submitting a credit request (*Acceptance of requests*). In the second step, the necessary documentation for the request is collected (*Collection of documents*), and in the third step the completeness of the documentation is checked (*Completeness check*). The following steps are checking the client's credit worthiness (*Credit worthiness check*) and checking the collateral (*Collateral check*). After that, the credit committee decides about acceptance of the credit request (*Credit committee*), and in the last step all the requests are reviewed (*Requirements review*).

The next step is determining the simulation parameters as it is previously described. First, the time of the activity execution in the process and the time of process instance generation are calculated using rules given in Table 1. After that, using the listed methods of statistical analysis, probability distribution function and their parameters are acquired as it is given in

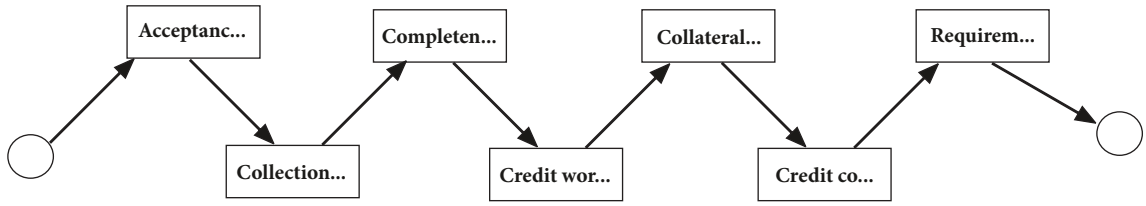

Figure 2: Process of credit requirement acquired using the Heuristic miner.

Table 3: Probability distribution functions used for task execution in the process.

Activity name	Distribution	Parameters (min)
Instance generation	Lognormal	mean=1.136; st. deviation=1.089
Acceptance of requests	Uniform	max=19.967; min=9.017
Collection of Documents	Weibull	shape=98.708; scale=3.290
Completeness check	Uniform	max=89.933; min=45.850
Credit worthiness check	Normal	mean=82.813; st. deviation=7.548
Collateral check	Uniform	max=74.383; min=20.133
Credit committee	Lognormal	mean=3.796; st. deviation=0.436
Requirements review	Uniform	max=14.917; min=5.033

Table 3. Parameters are acquired using MLE, MOME, and MGE methods. In Table 3, parameters are acquired using the MLE method.

Matlab is used for implementation of the DE algorithm. The process is first presented in Matlab using the SimEvents library. The activities in the process are presented using servers, and, using random number generators, the appropriate probability distribution functions of task execution in the process and process instance generation are represented. The number of servers presents the number of users executing tasks on each activity. For the observed process, these constraints are presented in the form of vectors of lower and upper bounds:

$$\alpha = [1 \;\; 1 \;\; 1 \;\; 1 \;\; 1 \;\; 3 \;\; 1]; \quad \beta = [10 \;\; 10 \;\; 10 \;\; 10 \;\; 10 \;\; 5 \;\; 10] \tag{16}$$

which means that the minimal number of resources on the first activity is 1, the maximal number is 10, as well as the second activity, and so on. On activity *Credit committee* the minimal number of resources is 3, and maximal number is 5. The units which do not comply with these constraints will be punished with penalties. The unit costs of the resources are as follows:

$$c = [1500 \;\; 2000 \;\; 2000 \;\; 2000 \;\; 1500 \;\; 3000 \;\; 1500] \tag{17}$$

The estimated value of constant a is 100, so it is necessary to minimize the function:

$$\min_{x_i} \left\{ 100 \cdot \sum_{i=1}^{n} k_i w_i (x_i) + \sum_{i=1}^{n} x_i c_i + \delta \right\} \tag{18}$$

This information about the process can be acquired from the process owner. Different strategies of the DE algorithm are implemented and 4 scenarios are observed.

Scenario 1. The first scenario entails the case when all the delays in the process are of the same importance. Those are cases when it is irrelevant on which activity users are waiting, all waitings in the process are unacceptable. In this scenario the coefficients are $k_1 = k_2 = \cdots = k_n = 1$.

Scenario 2. In the second scenario, some delays are more dangerous than others, which means that they are of greater significance. Such are the delays on the counter, which affect the user's satisfaction more than when a user is waiting for a call from the bank. In this scenario the coefficients are $k_1 = k_2 = 2, k_3 = k_4 \ldots = k_n = 1$.

Scenario 3. All of the activities in the process donot have to be directly linked to the users, and the waiting time on these activities does not have to be waiting time of the user in the process. Such delays need to be ignored, in the observed process that is the last activity *Requirements review*, where all the requests are reviewed. Because of that, the coefficients in this scenario are $k_1 = k_2 = \cdots = k_{n-1} = 1, k_n = 0$.

Scenario 4. The last scenario is the most complex and real. It represents a combination of the second and third scenario. In this scenario, not all delays of the same importance, and some of them do not need to be observed at all. The values of the coefficients are $k_1 = k_2 = 2$, $k_3 = k_4 \ldots = k_{n-1} = 1$, $k_n = 0$.

7.1. Discussion and Comparison of Results. Parameters of the algorithm were as follows: the number of generations was m=100, the size of the population was n=200, the crossover probability was CR=0.3, and the F factor was uniformly distributed between 0.2 and 0.8. The algorithm stops when there is no improvement of the best fitness function value in last 10 generations. The algorithm was run several times (10) and the average value of the number of generations

Table 4: Comparing different algorithm strategies (MLE method).

Scenario	Algorithm	Number of iterations	Best solution
Scenario 1	DE-1	66	[3 9 7 9 5 5 2]
	DE-2	73	[3 9 7 9 5 5 2]
	DE-3	20	[3 9 7 9 5 5 2]
	DE-4	28	[3 9 7 9 5 5 2]
	DE-5	46	[3 9 7 9 5 5 2]
	IDE	15	[3 9 7 9 5 5 2]
Scenario 2	DE-1	31	[3 10 7 9 5 5 2]
	DE-2	40	[3 10 7 9 5 5 2]
	DE-3	17	[3 10 7 9 5 5 2]
	DE-4	19	[3 10 7 9 5 5 2]
	DE-5	31	[3 10 7 9 5 5 2]
	IDE	11	[3 10 7 9 5 5 2]
Scenario 3	DE-1	38	[3 9 7 9 5 5 1]
	DE-2	46	[3 9 7 9 5 5 1]
	DE-3	17	[3 9 7 9 5 5 1]
	DE-4	19	[3 9 7 9 5 5 1]
	DE-5	37	[3 9 7 9 5 5 1]
	IDE	16	[3 9 7 9 5 5 1]
Scenario 4	DE-1	29	[3 10 7 9 5 5 1]
	DE-2	41	[3 10 7 9 5 5 1]
	DE-3	15	[3 10 7 9 5 5 1]
	DE-4	17	[3 10 7 9 5 5 1]
	DE-5	26	[3 10 7 9 5 5 1]
	IDE	10	[3 10 7 9 5 5 1]

and execution time was obtained. Figure 3 represents the convergence of the standard DE algorithm for different strategies and improved DE algorithm for first scenario. The x-axis represents the number of iterations, and the y-axis the value of the fitness function, which is given with (18). The figure illustrates that the starting value of the fitness function differs for different strategies, because the starting generation is randomly generated. It can be concluded that the algorithm succeeded in finding the optimal solution in an acceptable number of iterations, including the given constraints. Analyzing figure, it can be concluded that the improved DE algorithm outperforms standard DE algorithms. The improved DE algorithm succeeded to find the solution with the smallest number of iterations which can be seen in Table 4. The last column presents the best solution. It can be seen that all algorithms succeeded in finding the best solution, and the brute force algorithm was implemented to check whether the solution is optimal. Similar results are obtained using the other two methods for parameter estimation (MLE and MGE). These results are presented in Tables 5 and 6 and it can be noticed that the results are similar.

The brute force algorithm required about 6 days and 7 hours for one scenario, in order to test all possible combinations and to confirm that the optimal solution has been found. For the execution of the algorithm, a workstation with 8GB RAM memory and a 2.4 GHz processor was used. Comparing the best result from Scenarios 1 and 2, it can be concluded that a bottleneck exists on the second activity in the process. After

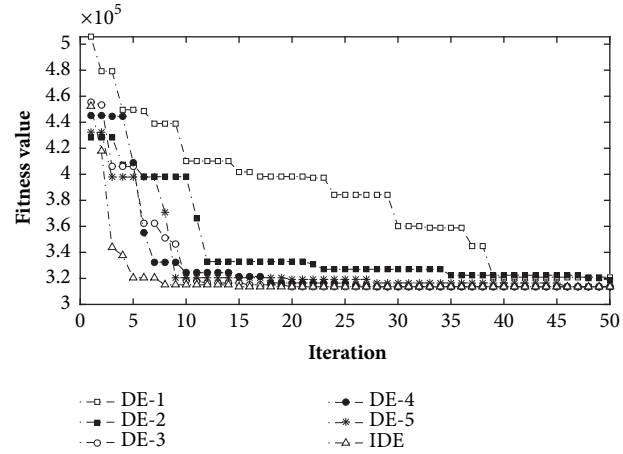

Figure 3: Comparison of algorithms for Scenario 1.

the significance of the delay on the first two activities in the process in Scenario 2 was doubled, by setting the values $k_1 = k_2 = 2$, the number of users on the second activity increased and a maximal number of users were allocated. The same case was with Scenario 4. On the other hand, in Scenarios 3 and 4 when the delay of the last activity in the process is disabled by setting $k_7 = 0$, the algorithm allocated a minimal number of users on that activity, because the expenses of the resources were minimal then.

Table 5: Comparing different algorithm strategies (MOME method).

Scenario	Algorithm	Number of iterations	Best solution
Scenario 1	DE-1	54	[3 9 7 9 5 5 2]
	DE-2	69	[3 9 7 9 5 5 2]
	DE-3	21	[3 9 7 9 5 5 2]
	DE-4	29	[3 9 7 9 5 5 2]
	DE-5	55	[3 9 7 9 5 5 2]
	IDE	5	[3 9 7 9 5 5 2]
Scenario 2	DE-1	42	[3 10 7 9 5 5 2]
	DE-2	49	[3 10 7 9 5 5 2]
	DE-3	27	[3 10 7 9 5 5 2]
	DE-4	28	[3 10 7 9 5 5 2]
	DE-5	42	[3 10 7 9 5 5 2]
	IDE	20	[3 10 7 9 5 5 2]
Scenario 3	DE-1	59	[3 9 7 9 5 5 1]
	DE-2	73	[3 9 7 9 5 5 1]
	DE-3	27	[3 9 7 9 5 5 1]
	DE-4	37	[3 9 7 9 5 5 1]
	DE-5	48	[3 9 7 9 5 5 1]
	IDE	23	[3 9 7 9 5 5 1]
Scenario 4	DE-1	47	[3 10 7 9 5 5 1]
	DE-2	37	[3 10 7 9 5 5 1]
	DE-3	23	[3 10 7 9 5 5 1]
	DE-4	27	[3 10 7 9 5 5 1]
	DE-5	31	[3 10 7 9 5 5 1]
	IDE	19	[3 10 7 9 5 5 1]

Table 6: Comparing different algorithm strategies (MGE method).

Scenario	Algorithm	Number of iterations	Best solution
Scenario 1	DE-1	61	[3 9 7 9 5 5 2]
	DE-2	54	[3 9 7 9 5 5 2]
	DE-3	21	[3 9 7 9 5 5 2]
	DE-4	23	[3 9 7 9 5 5 2]
	DE-5	40	[3 9 7 9 5 5 2]
	IDE	14	[3 9 7 9 5 5 2]
Scenario 2	DE-1	41	[3 10 7 9 5 5 2]
	DE-2	50	[3 10 7 9 5 5 2]
	DE-3	28	[3 10 7 9 5 5 2]
	DE-4	24	[3 10 7 9 5 5 2]
	DE-5	33	[3 10 7 9 5 5 2]
	IDE	21	[3 10 7 9 5 5 2]
Scenario 3	DE-1	57	[3 9 7 9 5 5 1]
	DE-2	60	[3 9 7 9 5 5 1]
	DE-3	31	[3 9 7 9 5 5 1]
	DE-4	33	[3 9 7 9 5 5 1]
	DE-5	55	[3 9 7 9 5 5 1]
	IDE	19	[3 9 7 9 5 5 1]
Scenario 4	DE-1	42	[3 10 7 9 5 5 1]
	DE-2	42	[3 10 7 9 5 5 1]
	DE-3	25	[3 10 7 9 5 5 1]
	DE-4	25	[3 10 7 9 5 5 1]
	DE-5	33	[3 10 7 9 5 5 1]
	IDE	21	[3 10 7 9 5 5 1]

Table 7: Validation results for credit requirements process.

Iteration	Model 1 (Y_1)	Model 2 (Y_2)	Model 4 (Y_3)	Y_1-Y_2	Y_1-Y_3
1	38823.79	254936.57	29973.53	-216112.78	8850.26
2	39295.19	280620.16	59629.32	-241324.97	-20334.13
3	38808.39	257944.03	32839.90	-219135.64	5968.49
4	38840.02	239623.39	29836.27	-200783.37	9003.75
5	39093.43	268929.24	77364.26	-229835.81	-38270.83
...	...	...	...	...	...
K_{ij}(%)	-	-	-	100	76.5

Table 8: Validation results for three different processes.

Process	Scenario	K_{12}(%)	K_{13}(%)
Credit requirements	Scenario 1	100	76.5
	Scenario 2	100	82
	Scenario 3	100	85
	Scenario 4	100	88
Electronic invoicing	Scenario 1	100	81
	Scenario 2	100	85.5
	Scenario 3	100	77
	Scenario 4	100	79.5
Document processing	Scenario 1	100	76.5
	Scenario 2	100	72.5
	Scenario 3	100	75
	Scenario 4	100	74

8. Model Validation

In order to show the success and applicability of the model, validation of the model with the current process model and comparison with the model from the literature was done. The method of validation and model comparison is shown on Figure 4. In the validation, the proposed model acquired through optimization and the second model which is currently used in the organization are compared. The comparison is done between the first model acquired through optimization and the third model from literature which is used for optimization. The difference between the first and third model is that statistical distribution is used for representing the activity duration in the first model, while the activity duration in the third model is expressed through the mean value. The models had a fixed resource allocation, while the events were randomly chosen from the log files and introduced in the processes. This way it was questioned how the exit from the process would behave if the resources were allocated according to the proposed allocation (model 1), according to the current allocation (model 2), and according to the allocation acquired using the model from literature (model 3). The provided comparison procedure of the models was implemented through 200 iterations for every single of the four scenarios. The acquired results are shown in Table 7. The table first presents the values of the fitness functions Y_1, Y_2, and Y_3, and then their differences. K_{ij} presents the percentage of how much model i was better than model j. It can be concluded from the table that model 1 was 100% better than model 2, while in 76.5% of the cases it was better than model 3. Similar results were acquired for the other scenarios. The improvement in model 1 compared to model 3 in Scenario 2 was 82%, in Scenario 3 it was 85%, and in Scenario 4 it was 88%. In order to show the generality of the approach, it was conducted on two more processes: electronic invoicing and document processing. The approach proved to be equally successful on all tested processes, which is shown in Table 8. As seen in the table, improvement was accomplished in all tested scenarios and ranges from 72.5% to 88%. This proves that the model is valid and that it gives better results than the model from literature.

9. Conclusion and Future Research

There are a lot of approaches dealing with the problem of resource allocation. Different techniques and algorithms are used for solving this problem, but little attention is paid to building a process model. In order to build a process model that accurately represents a real process, this paper uses a novel approach that combines process discovery algorithms and methods of statistical analysis. Process discovery algorithm enables getting processes in the form that they are executed in reality. Using methods of statistical analysis, probability distribution functions which can be used to describe the activity execution in the process have been acquired. The application of a stochastic simulation model

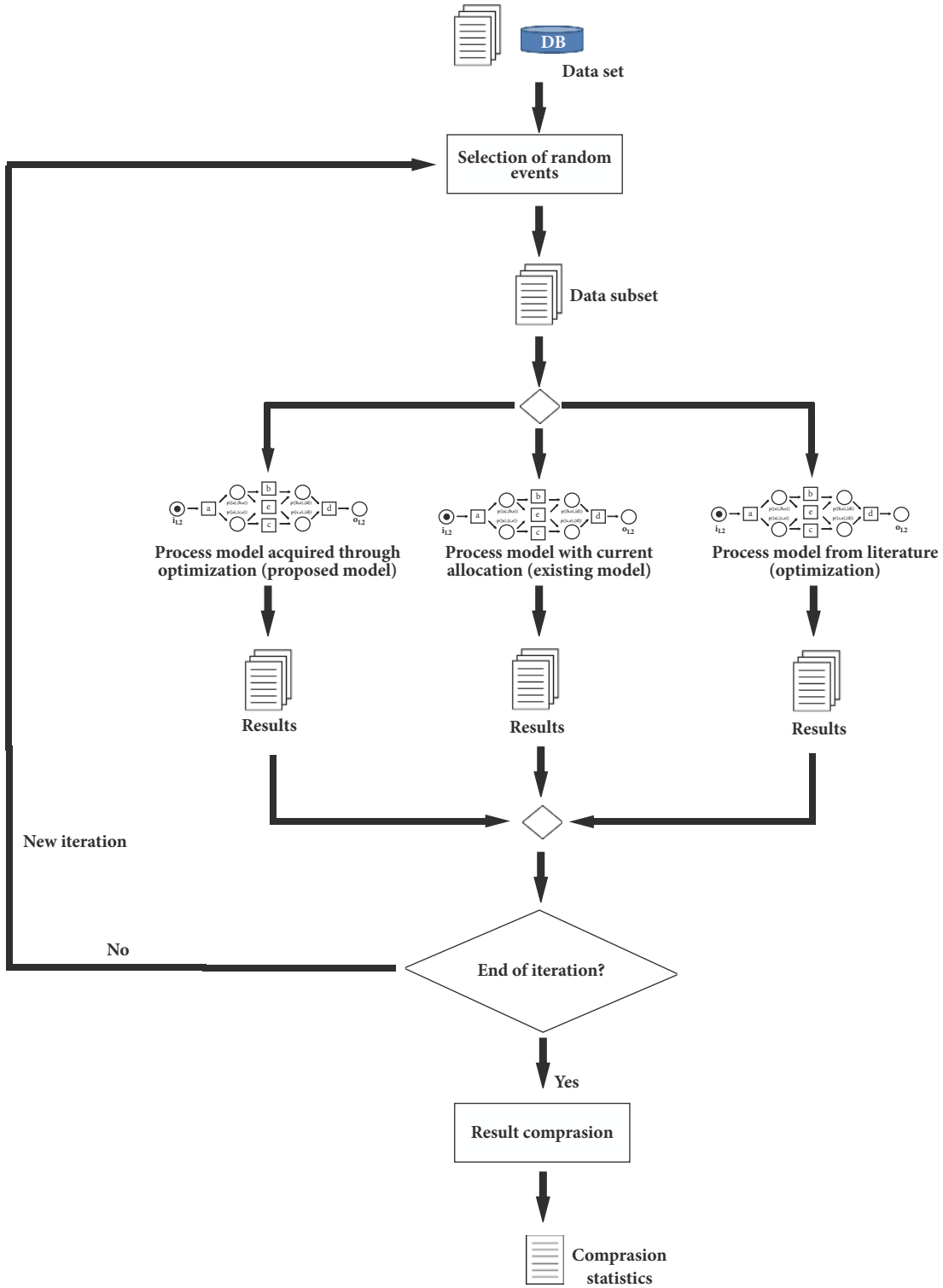

Figure 4: Model validation.

causes noise to appear in the fitness function. The differential evolution algorithm was used for finding the optimal human resources allocation with constraints. The algorithm was modified in order to include uncertainty in fitness function, and an improved differential evolution algorithm is proposed. The proposed algorithm had better performances than the standard DE algorithm on test functions as well as a real problem. Through different scenarios, it was shown how processes can be improved if the influence of the waiting time is not equal on all activities in the process, as well

as the case when it is necessary to exclude some activities. The presented approach is validated with the current process model and compared with the model from literature which has the activity duration in the process expressed through mean values. The proposed process model proved to be better in all scenarios for three different processes.

In this paper, the FIFO (first-in first-out) waiting queue has been observed. In the future work, it is intended to test other waiting principles such as LIFO (last-in first-out) and priority waiting queues. Also, process improvement will be analyzed using other optimization criteria such as resource utilization and average queue length.

Data Availability

The data used to support the findings of this study have been deposited in the online repository https://doi.org/10.4121/uuid:453e8ad1-4df0-4511-a916-93f46a37a1b5.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors would like to acknowledge the resource support to the Laboratory for Fusion of Artificial Intelligence in Bioinformatics and Biomedical Engineering (AIB) of the Faculty of Electrical Engineering in Sarajevo.

References

[1] W. M. P. van der Aalst, A. H. M. ter Hofstede, and M. Weske, "Business process management: a survey," in *Proceedings of the Bus. Process Manag*, vol. 2360, pp. 1–22, 2003.

[2] W. M. P. van der Aalst, "Business Process Simulation Revisited," *Enterp Organ Model Simul*, vol. 63, pp. 1–14, 2010.

[3] W. M. P. van der Aalst, J. Nakatumba, A. Rozinat, and N. Russell, "Business process simulation," in *Handbook on Business Process Management 1*, pp. 313–338, Springer, Berlin, Heidelberg, 2010.

[4] F. Guimei and H. Haijun, "Scenario-Based Stochastic Resource Allocation with Uncertain Probability Parameters," *Journal of Systems Science & Complexity*, vol. 30, no. 2, pp. 357–377, 2017.

[5] J. Hu, T. Homem-De-Mello, and S. Mehrotra, "Risk-adjusted budget allocation models with application in homeland security," *IIE Transactions*, vol. 43, no. 12, pp. 819–839, 2011.

[6] R. Levi and A. Radovanović, "Provably near-optimal LP-based policies for revenue man- agement in systems with reusable resources," *Operations Research*, vol. 58, no. 2, pp. 503–507, 2010.

[7] Y. Chen, R. Levi, and C. Shi, "Revenue Management of Reusable Resources with Advanced Reservations," *Production Engineering Research and Development*, vol. 26, no. 5, pp. 836–859, 2017.

[8] S. Dye, L. Stougie, and A. Tomasgard, "The stochastic single resource service provision problem," *Naval Research Logistics*, vol. 50, no. 8, pp. 869–887, 2003.

[9] P. Kouvelis and G. Yu, *Robust Discrete Optimization and Its Applications*, Kluwer Academic Publishers, 1997.

[10] S. Hartmann and D. Briskorn, "A survey of variants and extensions of the resource-constrained project scheduling problem," *European Journal of Operational Research*, vol. 207, no. 1, pp. 1–14, 2010.

[11] L. Borba and M. Ritt, "A heuristic and a branch-and-bound algorithm for the assembly line worker assignment and balancing problem," *Computers & Operations Research*, vol. 45, pp. 87–96, 2014.

[12] M. Vilà and J. Pereira, "A branch-and-bound algorithm for assembly line worker assignment and balancing problems," *Computers & Operations Research*, vol. 44, pp. 105–114, 2014.

[13] A. Gunawan and K. M. Ng, "Solving the teacher assignment problem by two metaheuristics," *International Journal of Information and Management Sciences*, vol. 22, no. 1, pp. 73–86, 2011.

[14] A. Djedović, E. Žunić, Z. Avdagić, and A. Karabegović, "Optimization of business processes by automatic reallocation of resources using the genetic algorithm," in *Proceedings of the 11th Int Symp Telecommun BIHTEL 2016*, 2016.

[15] J. Park, D. Seo, G. Hong et al., "Practical Human Resource Allocation in Software Projects Using Genetic Algorithm," in *Proceedings of the Twenty-Sixth Int Conf Softw Eng Knowl Eng*, pp. 688–694, 2014.

[16] J. Park, D. Seo, G. Hong, D. Shin, J. Hwa, and D. Bae, "Human Resource Allocation in Software Project with Practical Considerations," *International Journal of Software Engineering and Knowledge Engineering*, vol. 25, no. 01, pp. 5–26, 2015.

[17] I. E. Diakoulakis, D. E. Koulouriotis, and D. M. Emiris, "Resource Constrained Project Scheduling using evolution strategies," *Operational Research*, vol. 4, no. 3, pp. 261–275, 2004.

[18] J. Pantouvakis and O. G. Manoliadis, "A practical approach to resource - constrained project scheduling," *Operational Research*, vol. 6, no. 3, pp. 299–309, 2006.

[19] T. W. Liao, P. J. Egbelu, B. R. Sarker, and S. S. Leu, "Metaheuristics for project and construction management - A state-of-the-art review," *Automation in Construction*, vol. 20, no. 5, pp. 491–505, 2011.

[20] W. Van Der Aalst, T. Weijters, and L. Maruster, "Workflow mining: Discovering process models from event logs," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 9, pp. 1128–1142, 2004.

[21] A. K. de Medeiros, A. J. Weijters, and W. M. van der Aalst, "Genetic process mining: an experimental evaluation," *Data Mining and Knowledge Discovery*, vol. 14, no. 2, pp. 245–304, 2007.

[22] A. J. M. M. Weijters and J. T. S. Ribeiro, "Flexible heuristics miner (FHM)," in *Proceedings of the IEEE SSCI 2011 Symp. Ser. Comput. Intell. - CIDM 2011*, pp. 310–317, 2011.

[23] A. Djedovic, E. Zunic, and A. Karabegovic, "Model business process improvement by statistical analysis of the users' conduct in the process," in *Proceedings of the 2016 Int Multidiscip Conf Comput Energy Sci Split*, pp. 1–6, 2016.

[24] M. L. Delignette-Muller and C. Dutang, "Fitdistrplus: an r package for fitting distributions," *Journal of Statistical Software*, vol. 64, no. 4, pp. 1–34, 2015.

[25] Q. Liu and A. Ihler, "Distributed parameter estimation via pseudo-likelihood," in *Proceedings of the 29th International Conference on Machine Learning, ICML 2012*, pp. 1487–1494, Edinburgh, Scotland, UK, 2012.

[26] A. Di Pietro, L. While, and L. Barone, "Applying evolutionary algorithms to problems with noisy, time-consuming fitness functions," in *Proceedings of the 2004 Congress on Evolutionary Computation, CEC2004*, pp. 1254–1261, USA, June 2004.