Genetic Algorithm Approach for Casualty Processing Schedule

Marian Sorin Nistor
Universität der Bundeswehr
München,
85577 Neubiberg, Germany
sorin.nistor@unibw.de

Truong Son Pham
Universität der Bundeswehr
München,
85577 Neubiberg, Germany
son.pham@unibw.de

Stefan Wolfgang Pickl
Universität der Bundeswehr
München,
85577 Neubiberg, Germany
stefan.pickl@unibw.de

Abstract

Searching for an optimal casualty processing schedule can be considered a key element in the MCI response phase. Genetic algorithm (GA) has been widely applied for solving this problem. In this paper, it is proposed a GA-based optimization model for addressing the casualty processing scheduling problem (CPSP). It aims to develop a GA-based optimization model in which only a part of the chromosome (solution) involves in the evolutionary process. This can result in a less complex training process than previous GA-based approaches. Moreover, the study attempts to investigate two common objectives in CPSP: maximizing the number of survivals and minimizing the makespan. The proposed GA-based model is evaluated on two real-world scenarios in the Republic of Moldova, FIRE, and FLOOD. The paper suggests that GA models with a population size of 500 or smaller can be applied for MCI scenarios. The first objective can help many casualties receiving specialization treatments at hospitals.

1. Introduction

The response phase in a mass casualty incident (MCI) can be considered as the most important in mass casualty management (MCM). When a disaster happens, MCI responders must carry out several urgent responding actions, such as quickly assessing MCI scenes, mobilizing available resources, making an optimal schedule for processing casualties. The casualty processing that provides treatments to injuries and sends them to hospitals is the most significant stage among these actions. This may involve a sequence of tasks (also called operations) [1]. The first task is to rescue injuries if they are trapped (i.e., caught in the ruins of their houses), and provide stabilizing treatments if they are in unstable health conditions. Secondly, collecting and moving injuries to Casualty Clearing Station to classify into proper triage category. Finally, medical treatments are given to stabilize the victims during transportation to hospitals.

The casualty processing aims to minimize the number of victims. Thus, finding an optimal schedule for casualty processing can be considered a key element in achieving the objective. Generally, two common objectives are often used to search for an optimal casualty processing schedule, such as maximizing the number of survivals and minimizing the total processing time (makespan). In practice, an optimal schedule obtained with the first objective is also relatively satisfied with the second objective and vice versa [1, 2]. This means that minimizing the makespan can result in many casualties sent to hospitals on time. Maximizing the total number of survivals also leads to many casualties receiving necessary treatments at hospitals within their waiting time.

Recently, mathematically modeling the CPSP has been received significant attention from the research community. Many studies have attempted to develop optimization algorithms to search for an optimal schedule for processing casualties, such as GA, NSGA-II, deep reinforcement learning (DRL) [1–6]. First, CPSP is modeled as a flexible Job Shop schedule problem (FJSSP) [1, 2], and then optimization-based algorithms are employed to search for an optimal schedule. Amongst these algorithms, GA-based optimization techniques are widely applied for this task. However, the GA chromosomes (solutions) in previous work [2, 3] are complex. More importantly, all parts of the chromosome are involved in the evolutionary process resulting in a complexity of the training process.

In this work, it is proposed a GA approach for solving the casualty processing scheduling problem. The approach is in two phases: (1) CPSP is modeled as FJSSP, and (2) define the GA chromosome (solution) and apply GA to evolve on the part of the chromosome. The proposed GA model is evaluated on two real-world MCI scenarios, FIRE and FLOOD, in the Republic of Moldova. The optimal schedule is searched for these scenarios concerning the two common objectives, maximizing the number of survivals (the first objective)

URI: https://hdl.handle.net/10125/79499
978-0-9981331-5-7
(CC BY-NC-ND 4.0)

and minimizing makespan (the second objective).

The remainder of the paper is presented as follows. Sections 2 and 3 describe shortly some backgrounds on GA and FJSSP for understanding the followed sections. Following this, the proposed GA approach for CPSP and experimental descriptions are presented in Sections 4 and 5. Finally, Sections 6 and 7 provide the resulting discussion, conclusion, and future direction for the paper.

2. Background

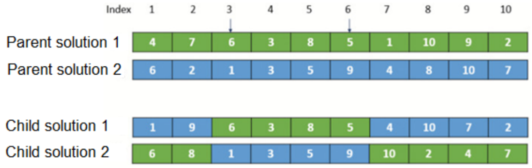
In this section, the background of this paper is presented by briefly introducing the GA followed by the FJSSP.

2.1. Genetic Algorithm

It is a metaheuristic inspired by the process of natural selection that belongs to the larger class of evolutionary algorithms (EA). Genetic algorithms are commonly used to generate high-quality solutions to optimization and search problems by relying on biologically inspired operators such as mutation, crossover, and selection [7]. The evolutionary process iterates for several generations until an optimal solution is found. The evolutionary process starts with an initial population of individuals in the first generation. Each individual is a candidate solution for the schedule of the casualty processing. At every generation, several solutions are selected from the current population (called parent solutions). These parent solutions are then operated by operators, such as crossover and mutation, to create new candidate solutions (called children). If these children are satisfied with the constraints of GA, they are placed into a pool for creating a new population in the next generation. A fitness function is employed during the evolutionary process to guide the process to reach the optimal target. In other words, fitness value is used to evaluate the performance of individuals. For a given solution, the better the fitness value, the better performance it can yield, and the more likely it can be selected as a parent to produce new individuals for the next generation. The basic GA components such as crossover, mutation, selection, and fitness function are introduced in the following sub-sections.

Crossover operator:

To create new solutions by the crossover operation, two-parent solutions, (PS_1) and (PS_2), are chosen by the selection operator concerning their fitness values. These solutions can be processed by the OX crossover operator that swaps a part of PS_1 with a part of PS_2. As a result, two new candidate solutions are created. If these child candidates are satisfied with the constraints of the solution, they are new children, CS_1 and CS_2, in the next generation.

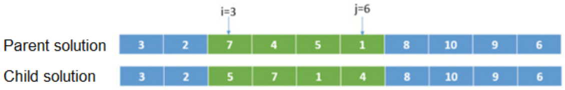


Index	1	2	3	4	5	6	7	8	9	10
Parent solution 1	4	7	6	3	8	5	1	10	9	2
Parent solution 2	6	2	1	3	5	9	4	8	10	7
Child solution 1	1	9	6	3	8	5	4	10	7	2
Child solution 2	6	8	1	3	5	9	10	2	4	7

Figure 1: An illustration of the OX crossover operator

Figure 1 shows two parents and their children yielded by the XO crossover operator. A set of integer numbers describes the solutions. Depending on a specific problem, a real value can be used instead of an integer. The operator swaps two random segments on the parents PS_1 and PS_2 to create two new solutions CS1 and CS_2. In the figure, the first two rows represent two parent solutions, and the second is the resulting solutions of children.

Mutation operator: A parent solution can be mutated by itself. The mutation process randomly selects several items between two points on the parent and then randomly swap them to create a new child solution. In Figure 2, four elements from $i = 3$ to $i = 6$ are randomly swapped to create a child solution. Crossover and mutations are operated to create the next generation of solutions until the convergence criteria met. When applying GA for CPSP, the constraints of the solution defined in Section 4 is applied to check wherever a new candidate satisfied or not.



Parent solution	3	2	7	4	5	1	8	10	9	6
Child solution	3	2	5	7	1	4	8	10	9	6

Figure 2: An illustration of the mutation operator

Selection operator: Chromosomes in a generation can be selected based upon their fitness value. The chromosomes satisfied with the fitness criteria are more likely to be selected in each newer generation. Several selection criteria include roulette wheel selection, rank-based fitness assignment, elitism, and tournament selection. Amongst these criteria, tournament selection is widely used and is also employed in this work.

Fitness function: In this work, two fitness functions are designed based on the two objective functions mentioned in Section 4. They are maximizing the number of survivals and minimizing the makespan. These fitness functions are employed in two separate optimization processes.

2.2. Flexible Job-Shop Scheduling Problem

The schedule of the casualty processing can be modeled as a FJSSP. Some definitions of a classical FJSSP can be found in earlier publications [2,4]. FJSSP is briefly formulated as follows [2,4]:

- Given a set of n jobs $\{J_i, 1 \leq i \leq n\}$ and a set of m machines $\{M_j, 1 \leq j \leq m\}$.
- Each job J_i is represented by a sequence of n_i operations $O_{i,j}$.
- Each operation $O_{i,j}$ can be processed on a subset $\{M_{i,j}, 1 \leq i \leq n\ 1 \leq j \leq n_i\}$ of the available machines.
- The time $t_{i,j,k}$ represents a fixed and predetermined processing time for the operation $O_{i,j}$ on each machine.

It is essential to mention that the FJSSP works under two assumptions. First, all machines are available initially, and second, one machine can handle only one job at a time [4].

Based on the definition above, it is modeled the CPSP as an FJSSP in Section 4 by representing casualties as the jobs, medical teams as the machines, treatment tasks as the operations, and the treatment duration as the processing time of the operation. The goal is to optimize the model with GA by minimizing the makespan or maximizing the total number of surveillance. Compared to FJSSP, CPSP is more complex to model and solve because it is very challenging to determine the exact treatment time initially, and some uncertainties may occur during the process [4]. Therefore, it is one of the research directions to be able to handle the uncertainties.

3. Related Work

Researchers often model a CPSP as a FJSSP. They then solve the problem with meta-heuristic algorithms, such as GA and Variable Neighborhood Descent (VND), by minimizing the makespan and the mortality. The GA-based approach is the most common one used in literature. This section discusses some of the most recently published and well-known papers in this field.

Wilson et al. [1] proposed a novel combinatorial multi-objective optimization model of a CPSP by employing a scheduling approach. The Variable Neighborhood Descent is then implemented to optimize the model, and a range of potential problems is used to evaluate the model to confirm its effectiveness and robustness. The results showed that, compared to the constructive heuristic Φ, VND is less effective regarding the makespan. However, its resulting total number of arrivals during the first hour is higher [1].

In a paper recently published, Viana et al. [3] proposed a modified GA for JSSP by generalizing the concept of massive local search to develop a new multi-crossover operator. It aims to optimize the makespan in JSSP. The authors evaluated their proposed method with three different case studies. The results showed that mXLSGA is a robust method and performs better than other GA methods with an even better convergence rate [3].

Also, by improving the classical GA, Zhang et al. [8] presented a novel approach to support a bi-directional scheduling problem. The original GA was enhanced to handle both the key and typical jobs. The authors also applied a rolling window for rescheduling under the effects of local dynamic disruptions. To do that, they first classify the disorders into three levels: First-level events require a complete rescheduling, a local rescheduling is enough for the second-level one, where a right shift rescheduling is necessary for the third-level events. The approach was evaluated with simulations and could show its effectiveness in this domain.

4. Genetic Algorithm Approach for CPSP

This section provides some definitions of the casualty processing scheduling problem and how to model it as a FJSSP. Then, how to apply the GA-based optimization method is described to search for an optimal schedule of casualty processing.

4.1. Definitions

Definition 1: Casualty Processing Scheduling Problem. The symbols to present notions for a given MCI scenario are introduced. The details are as follows,

- C_i: i-th casualty, $1 \leq i \leq n$, where n is the total number of casualties.
- $O_{i,j}$: the operation j-th on the casualty C_i, $1 \leq i \leq n, 1 \leq j \leq n_i$.
- L_i: the triage level of casualty C_i, $L_i \in \{P_3, P_2, P_1, D/M\}$. D and M refer to dead and missing situations respectively.
- R_k: the responding team k, $1 \leq k \leq m$, where m is the number of responding teams.
- n_o: the maximum number of operations that can be applied for any casualties if

needed. The possible operations can be presented by $O = \{rescue, \ first - aid\,treatment, \ transport\ to\ hospitals\}$

- $t_{i,j,k}$: the time duration for the responding team R_k needs to complete j-th operation on the casualty C_i.

Definition 2: Constrains on the casualty processing schedule. A set of constraints on the casualty processing are also defined as follows,

1. Operations on casualties are performed following the order of *rescue*, *first-aid treatment*, and *transport to hospitals*.

2. At a time, a responding team R_k can process only one operation $O_{i,j}$ on one casualty C_i.

3. The operations *rescue, first-aid treatment, transport to hospitals* can be performed on any casualties. Some casualties may not need some treatments, such as rescue and first-aid treatment. If so, the processing time is equal to zero.

4. Responding teams are assumed to consist of doctors, nurses, ambulances, firemen, and police. Thus, responding teams can perform any operations on any casualties.

5. If a responding team R_k performs the operation $O_{i,j}$ on the casualty C_i after finishing the operation of transporting casualty C_q to a hospital, the starting time of $O_{i,j}$ should be added to the time duration R_k come back from the hospital to the scene.

4.2. GA representation

As mentioned before, the schedule of the casualty processing can be modeled as a Flexible Job Shop Scheduling. GA is then employed to search for an optimal solution. This section presents how to formulate CPSP in the form of the GA-based solution, and how to use the GA-based optimization to solve CPSP.

Chromosome representation: For a given MCI scenario with n casualties, the maximum number of operations on a casualty n_o, and m responding teams, the given problem in a format of the GA-based solution is represented as follows.

- Chromosome: a sequence of operations from n casualties in a casualty processing schedule can be presented as a GA chromosome. Each chromosome is denoted by two matrixes: a Gene matrix (G) of n x n_o, and a Position matrix (P) with the same size.

- Gene matrix (G): each row represents each casualty, and each column represents operations in each casualty. An item $g_{ij} \in G$ is a responding team R_k that carries out the operation $O_{i,j}$.

- Position matrix (P): The format of P is the same as G. However, each element $p_{ij} \in P$ represents the processing order of operation $O_{i,j}$ (the j-th operation on the casualty C_i) in the responding team R_k.

- G is evolved by GA while P can be generated by traversing on a resulting G concerning the waiting time of casualties and the order of the operations in casualties. The shorter waiting time a casualty has, the sooner operations of the casualty are scheduled. Only a part of a chromosome, the G matrix, involves in the GA evolutionary process. Thus, the complexity of this evolution process is less than that of the GA model introduced in [2].

An example is taken to explain more about the proposed chromosome. Let a MCI scenario has the number of casualties $n = 4$, the maximum number of operations in each casualty $n_o = 3$ and the number of responding teams $m = 3$. The gene (G) matrix and the Position (P) matrix for this scenario can be described as follow,

$$Gene(G) = \begin{bmatrix} \mathbf{1} & 2 & 2 \\ 2 & \mathbf{1} & \mathbf{1} \\ \mathbf{1} & \mathbf{1} & \mathbf{1} \\ 2 & 2 & \mathbf{1} \end{bmatrix} \quad Position(P) = \begin{bmatrix} \mathbf{1} & 3 & 5 \\ 1 & \mathbf{3} & \mathbf{5} \\ \mathbf{2} & \mathbf{4} & \mathbf{6} \\ 2 & 4 & \mathbf{7} \end{bmatrix}$$

The value of $g_{2,1} = 1$ indicates that the operation $O_{2,1}$ of the casualty C_2 is processed by the responding team R_1. Corresponding to $g_{2,1} \in G$, $p_{2,1} = 4$ indicates that the operation $O_{2,1}$ of casualty C_2 has a processing order of 4 in the responding team R_1.

Generate P matrix: P matrix does not involve in the GA evolutionary process. Instead, this matrix is created when a new candidate G is generated. In other words, once a G matrix is yielded by the crossover or mutation process, the items of G are traversed to determine the processing order of each operation in responding teams. The values of these processing orders are used to create P. Traversing G is followed two rules such as (1) the shorter waiting time a casualty has, the sooner it is visited; (2) in a casualty, the operation order is the *rescue*, *first-aid treatment*, and *transport to suitable hospitals*.

Evaluate solutions: Once G and P are created, the processing time, $t_{i,j,k}$, for all operations of casualties, and the completed time points of all casualties are calculated. If a responding team R_k processes $O_{i,j}$ of C_i after sending C_p to a hospital, the starting time point of $O_{i,j}$ should be added the traveling time of R_k from the hospital to the scene. The fitness value of solutions can be estimated using the total number of casualties arriving hospitals within their waiting time when using the first objective, and the completed time point of the last casualty arriving hospitals (makespan) when using the second objective.

5. Experiments

This section presents two real-world MCI scenarios and their corresponding processed data tables. Then, two main experiments and the parameter settings for the GA algorithm and the two MCI scenarios are described.

5.1. Data scenarios

Table 1: Processed data for the first scenario

Casualty	Triage	Processing time			Waiting
	level	O_1	O_2	O_3	time
1	3	5	5	33.2358	600
2	3	0	5	33.2358	600
3	3	5	0	33.2358	600
4	3	0	5	33.2358	600
5	3	7	0	33.2358	600
6	3	0	5	33.2358	600
7	3	5	6	33.2358	600
8	3	0	5	33.2358	600
9	3	5	6	33.2358	600
10	3	0	7	36.6296	600
11	3	5	0	36.6296	600
12	3	0	5	36.6296	600
13	3	5	0	36.6296	600
14	3	0	6	36.6296	600
15	3	5	0	36.6296	600
16	3	0	7	36.6296	600
17	3	6	6	36.6296	600
18	3	5	0	36.6296	600
19	3	0	7	36.6296	600
20	3	5	0	36.6296	600
21	2	20	10	33.2358	120
22	2	10	8	33.2358	120
23	2	15	5	33.2358	120
24	2	22	15	36.6296	120
25	2	14	8	36.6296	120
26	M	120	30	33.2358	720
27	M	120	30	33.2358	720

Table 2: Processed data for scenario 2

Casualty	Triage	Processing time			Waiting
	level	O_1	O_2	O_3	time
1	3	0	5	39.2057	600
2	3	6	0	39.2057	600
3	3	0	6	39.2057	600
4	3	6	0	39.2057	600
5	3	0	7	39.2057	600
6	3	7	0	39.2057	600
7	3	8	5	39.2057	600
8	3	0	0	39.2057	600
9	3	5	6	39.2057	600
10	3	0	0	39.2057	600
11	3	5	5	39.2057	600
12	3	5	0	104.5747	600
13	3	0	6	104.5747	600
14	3	5	6	104.5747	600
15	3	7	0	104.5747	600
16	3	0	5	104.5747	600
17	3	7	0	104.5747	600
18	3	0	5	104.5747	600
19	3	5	0	104.5747	600
20	3	0	6	115.1097	600
21	3	5	7	115.1097	600
22	3	0	0	128.4802	600
23	3	5	5	128.4802	600
24	3	0	0	128.4802	600
25	3	0	5	128.4802	600
26	3	5	0	128.4802	600
27	3	0	6	128.4802	600
28	3	6	0	128.4802	600
29	3	0	5	128.4802	600
30	3	6	0	128.4802	600
31	3	0	5	128.4802	600
32	3	7	0	128.4802	600
33	3	0	6	128.4802	600
34	3	8	0	128.4802	600
35	3	0	5	128.4802	600
36	3	5	5	39.2057	480
37	3	5	0	46.4784	480
38	3	0	8	39.609	480
39	3	6	0	46.4784	480
40	3	8	6	39.609	480
41	2	0	5	4.0013	120
42	2	10	12	4.0013	120
43	2	0	0	39.2057	120
44	D	10	15	4.0013	720

FLOOD scenario: It has been raining heavily for the past few days in the North of Republic of Moldova. Three villages are flooded because the Costesti-Stanca Lake, Reservoir Lake on Prut River, exceeded water level considered dangerous according to the representatives from the Hydro-technical Node. The affected villages are Costesti, Proscureni and

Duruitoarea. Twenty persons* are injured because of the powerful flood streams in the villages, and five** more are caught in the ruins of their houses, and two*** others are reported missing[1]. The objective is to send the red code victims to the nearest available hospitals with Traumatology specialization and the yellow code victims to the nearest available hospitals for acute medical treatment.

FIRE scenario: A fire started from burning the dried vegetation went out of control and affected the outskirt area of Floresti City. The firefighters from the city tried to contain the fire from the extension, but unfortunately, due to the limited intervention resources, several houses are reported to be on fire. Thirty-five* persons are suffering from smoke intoxication and require medical attention, five* persons suffered from second-degree burns, three** persons suffered from fourth-degree burns that are in a critical state and need immediate surgical treatment, and one*** person is declared dead[1]. The objective is to send the red code victims to the nearest available hospitals with surgical specialization; the second priority is given to the victims with burns from the yellow code. Finally, transport the yellow code victims intoxicated with smoke to the nearest available hospitals with pneumology specialization.

Processed data tables for two scenarios: Based on the triage code and specialization treatments of casualties, and available resources of nearest hospitals, processed data tables for these scenarios are created as shown in Tables 1 and 2. The processing time for rescue O_1 and first-aid treatment $O2$ can vary in the range of $[5, 30]$ minutes. The transport time to hospitals depends on the distance between the scene and the target hospital. In this work, the speed of ambulances on all routes to hospitals is assumed to be the same, and the resources of hospitals can have availability of 30% every. Therefore, only hospitals that satisfy the specialization treatments and can be reached within 2 hours are chosen to send casualties. Each casualty gets assigned a waiting time depending on the triage code. Waiting time is the maximum time a casualty can wait for a specialization treatment at hospitals before casualty is dead.

5.2. Experimental Settings

Two main experiments for evaluating the proposed GA model on the two real-world scenarios are designed. The first is to investigate different GA settings (population size) for the CPSP. This aims to find out a proper GA settings for deeper investigation later. In this experiment, five different population sizes of 10, 100, 200, 500, and 1000 are examined. Five different GA models evolves using the objective of maximizing the number of survivals over 300 generations. The running time, survivals, and makespan are reported. The second experiment is to evaluate the performance of the GA model with two fitness functions on the two scenarios. Four aspects of the GA performance are observed: the number of survivals, makespan, running time, and also Gantt chart. Table 3 presents the brief information of the two scenarios and the parameter setting of GA.

Table 3: Parameter settings for scenarios and GAs

No	Parameters	Values	Notes
	Scenario 1		
1	No of casualties	27	Include
2	No of responding teams	4	missing
3	No of treatment operations	3	
4	Chromosome length	81	27x3
	Scenario 2		
1	Number of casualties	44	Include
2	No of responding teams	8	deaths
3	No of treatment operations	3	
4	Chromosome length	132	44x3
	Optimiztion parameters		
1	Crossover rate	0.9	
2	Mutation rate	0.15	
3	Tournament size	5	
4	No of generations	300	
5	Population size	500[2]	

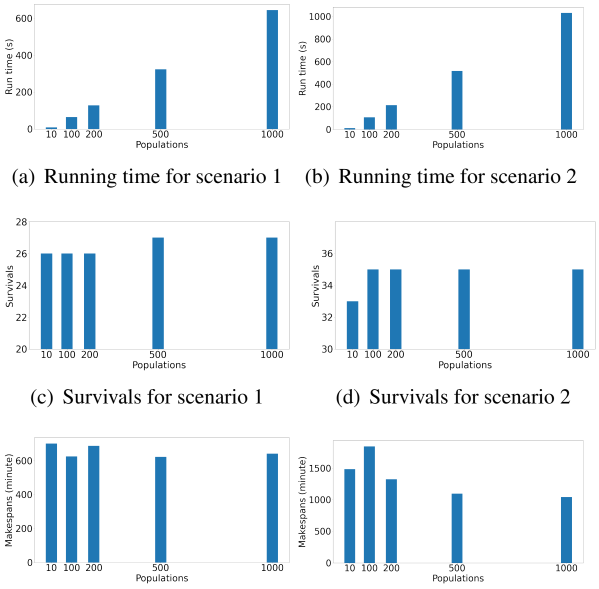
6. Results and Discussion

Figure 3 illustrates the performance of the GA model on the five settings of the population size on the two scenarios. Sub-figs 3(a) and 3(b) plots running time (in a minute) against population size. It can be seen that the running time has a linear correlation with population size. In incident response, the running time of the optimization models should be small enough. Therefore, the GA models with a population size up to 500 are feasible for MCI because their running time is short enough. Sub-figs 3(c), 3(d), 3(e) and 3(f)show that the larger population size is, the better quality (larger number of survivals and smaller makespan) resulting schedules have. Thus, the setting of 500 individuals can be considered as one of the best choices for these MCI scenarios, and used for the second experiment.

Tables 4 and 5 present the results for the two scenarios when using the two objectives. With the

[1] The notation *, **, and *** are used to refer the yellow triage code, red triage code, red/black triage code respectively.

[2] In the first experiment, GA with different population sizes of 10, 100, 200, 500, and 1000 is investigated.



(a) Running time for scenario 1 (b) Running time for scenario 2

(c) Survivals for scenario 1 (d) Survivals for scenario 2

(e) Makespan for scenario 1 (f) Makespan for scenario 2

Figure 3: Performance on five different population sizes

Table 4: Results on scenario 1 with two objectives

Obj	Res. team	Survival casualty	Total
1	0	1, 6, 11, 14, 15, 18, 24	27 survivals 658 minutes
	1	3, 9, 10, 13, 16, 19, 20	
	2	0, 4, 5, 7, 21, 22, 26	
	3	2, 8, 12, 17, 23, 25	
2	0	0, 3, 7, 8, 9, 18	25 survivals 581 minutes
	1	4, 5, 11, 21, 26	
	2	2, 6, 12, 15, 16, 19, 20	
	3	1, 10, 13, 14, 17, 23, 25	

Table 5: Results on scenario 2 using two objectives

Obj	Res. team	Survival casualty	Total
1	0	7, 12, 25, 38, 39	35 survivals 1103 minutes
	1	21, 30, 36	
	2	0, 2, 3, 18, 41	
	3	24, 29, 35	
	4	6, 10, 13, 20, 42, 43	
	5	4, 9, 17, 19, 40	
	6	8, 11, 15, 37	
	7	1, 5, 14, 26	
2	0	5, 12, 16	32 survivals 980 minutes
	1	0, 1, 2, 4, 14, 39	
	2	6, 13, 22, 35	
	3	17, 24, 37, 40	
	4	7, 9, 15, 38, 41	
	5	3, 8, 10, 23, 42	
	6	18, 21	
	7	11, 19, 36	

Table 6: G matrix for scenario 1 with survival objective

Casualty	O_1	O_2	O_3	Casualty	O_1	O_2	O_3
1	3	3	2	15	2	0	0
2	3	0	0	16	2	1	0
3	1	0	3	17	3	0	1
4	3	1	1	18	3	0	3
5	2	0	2	19	2	2	0
6	0	0	2	20	0	2	1
7	1	2	0	21	1	1	1
8	3	2	2	22	2	2	2
9	0	3	3	23	3	0	2
10	1	1	1	24	1	1	3
11	0	2	1	25	0	3	0
12	3	2	0	26	2	1	3
13	1	0	3	27	3	3	2
14	3	2	1				

first objective, the resulting schedules can save the life of 27 and 35 victims from scenarios 1 and 2, respectively. The number of survivals reduces to 25 and 32 for the first and second scenarios separately. On the other hand, the makespan for both scenarios decreases considerably when applying the second objective, 1.3 and 2.0 hours on scenarios 1 and 2, individually. Therefore, concerning the requirement of scenarios, the first objective is more preferable. The details of the G matrices when using the first objective on two scenarios are also reported in Tables 6 and 7.

However, the results in Tables 4 and 5 also raise the question of why optimizing with the first objective producing a higher makespan still achieves a larger number of survivals than that with the second objective. This question can be answered by using the information from Gantt charts in Figure 4. It can be seen from Figures 4 that most of the responding teams complete their task very soon while few of them last for a long period. This means that the optimizer with the first objective can save many survivals while accepting few cases being processed late. On the other hand, the completed time points on every responding team seem to be very similar, as shown in Figure 5. In other words, the makespan objective aims to finish all tasks as soon as possible. Thus, operations can be assigned equally to all responding teams resulting in many deaths.

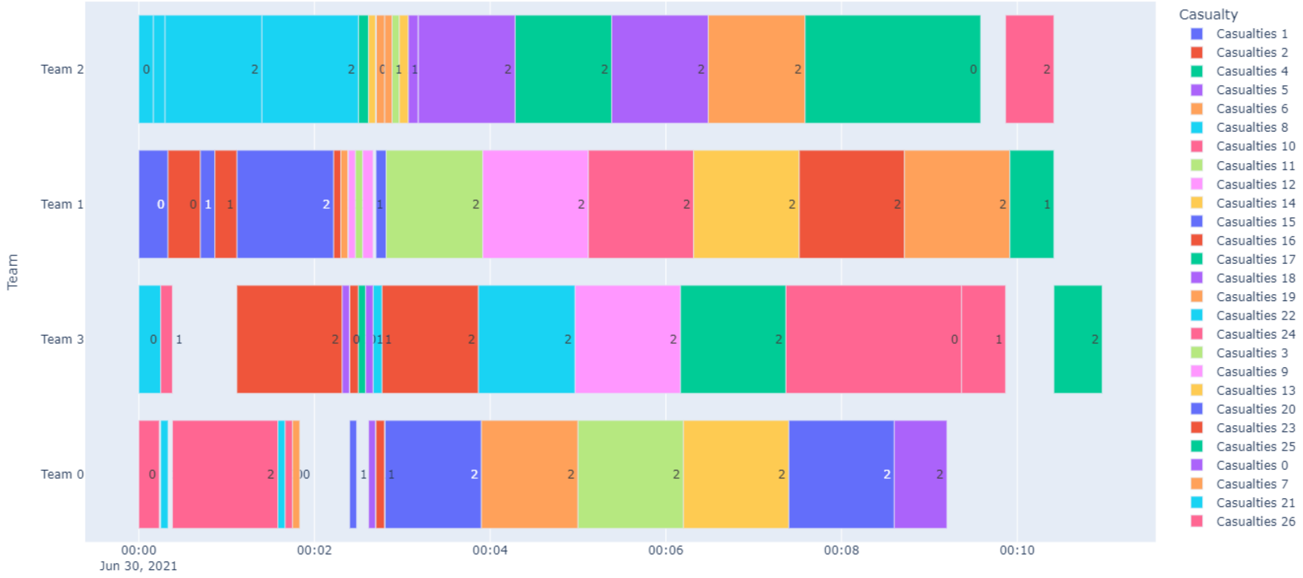


Figure 4: Gantt chart for scenario 1 with survival objective

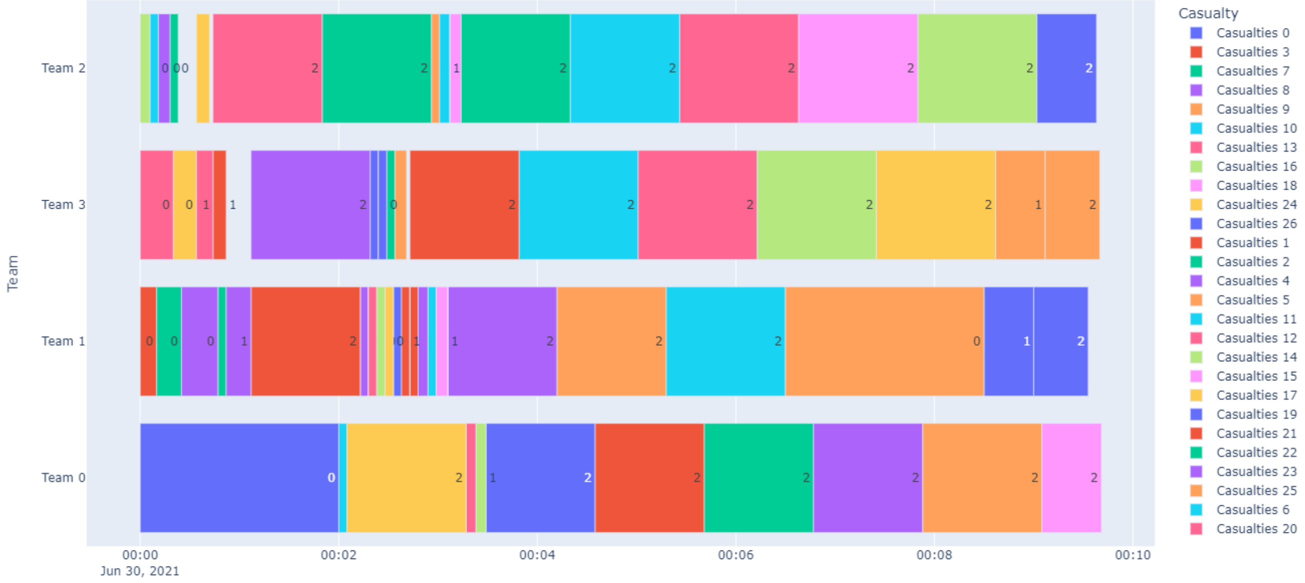


Figure 5: The Gantt chart for scenario 1 with makespan objective

Table 7: G matrix for scenario 2 with survival objective

Casualty	O_1	O_2	O_3	Casualty	O_1	O_2	O_3
1	4	5	2	23	4	2	2
2	0	4	7	24	6	5	5
3	0	6	2	25	7	3	3
4	2	2	2	26	5	1	0
5	4	2	5	27	5	3	7
6	2	4	7	28	6	1	0
7	5	5	4	29	1	6	5
8	1	0	0	30	4	7	3
9	1	2	6	31	5	1	1
10	3	2	5	32	7	5	2
11	1	4	4	33	3	4	7
12	3	0	6	34	7	2	6
13	2	1	0	35	1	7	7
14	2	1	4	36	3	1	3
15	2	7	7	37	4	7	1
16	2	7	6	38	1	7	6
17	6	3	6	39	3	3	0
18	2	1	5	40	1	3	0
19	2	4	2	41	7	1	5
20	2	4	5	42	5	5	2
21	4	5	4	43	5	4	4
22	7	7	1	44	3	1	4

7. Conclusion and Future Work

This paper presents the GA-based optimization model for solving the casualty processing scheduling problem from two real-world MCI scenarios in the Republic of Moldova, FIRE, and FLOOD.

The novelty of the proposed GA-based model is that only a part of the chromosome (solution) involves in the GA evolutionary process, which results in less complexity of the training process than the previous GA-based approach. Two experiments are carried out. The first is to examine the GA-based model on five different population sizes to observe the GA performance. The second is to extensively investigate the performance of the proposed GA-based optimization with two common objectives, and evaluate it on a metric of survivals, makespan, and Gantt charts.

The experimental results suggest that the population size of 500 or smaller can be feasible for solving CPSP. Moreover, the objective of maximizing the number of survivors is more preferable to minimizing makespan in saving the lives of casualties. The task of dealing with a dynamic CPSP is considered in future work.

8. Acknowledgments

This research was sponsored by the NATO Science for Peace and Security Programme under grant SPS MYP G5700.

References

[1] D. T. Wilson, G. I. Hawe, G. Coates, and R. S. Crouch, "A multi-objective combinatorial model of casualty processing in major incident response," *European Journal of Operational Research*, vol. 230, no. 3, pp. 643–655, 2013.

[2] X. Chu, Q.-Y. Zhong, and S. G. Khokhar, "Triage scheduling optimization for mass casualty and disaster response," *APJOR*, vol. 32, no. 06, p. 1550041, 2015.

[3] M. S. Viana, O. Morandin Junior, and R. C. Contreras, "A modified genetic algorithm with local search strategies and multi-crossover operator for job shop scheduling problem," *Sensors*, vol. 20, no. 18, p. 5440, 2020.

[4] M. K. Amjad, S. I. Butt, R. Kousar, R. Ahmad, M. H. Agha, Z. Faping, N. Anjum, and U. Asgher, "Recent research trends in genetic algorithm based flexible job shop scheduling problems," *Mathematical Problems in Engineering*, vol. 2018, 2018.

[5] G. Al Aqel, X. Li, and L. Gao, "A modified iterated greedy algorithm for flexible job shop scheduling problem," *CJME*, vol. 32, no. 1, pp. 1–11, 2019.

[6] S. Ji, Y. Zheng, Z. Wang, and T. Li, "A deep reinforcement learning-enabled dynamic redeployment system for mobile ambulances," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 3, no. 1, pp. 1–20, 2019.

[7] M. Mitchell, *An introduction to genetic algorithms*. MIT press, 1998.

[8] H. Zhang and Y. Zhang, "A discrete job-shop scheduling algorithm based on improved genetic algorithm," *Int. J. Simul. Model*, vol. 19, pp. 517–528, 2020.