Resource Dispatch Optimization for Firefighting Based on Genetic Algorithm*

Marina A. Matos[1[0000−0001−5387−8771]], Ana Maria A. C. Rocha[1[0000−0001−8679−2886]], Lino A. Costa[1[0000−0003−4772−4404]], and Filipe Alvelos[1[0000−0002−1851−4339]]

ALGORITMI Research Centre / LASI, University of Minho, Campus de Gualtar, 4710-057 Braga, Portugal
mmatos@algoritmi.uminho.pt, {arocha,lac,falvelos}@dps.uminho.pt

Abstract. The number of forest fires has increased in recent years. Rising ambient temperatures and rising demographics are the main drivers of these disasters. Optimization has been widely applied in forest firefighting problems, allowing improvements in the effectiveness and speed of firefighters actions. In this work, a resource dispatch problem for forest firefighting (involving 7 resources to extinguish 20 ignitions) is presented. The main goal is to minimize the total burned area caused by the ignitions. To solve this work, a genetic algorithm (GA) adapted to this problem was used. Furthermore, a statistical analysis was carried out among several GA operators, crossover, mutation and selection, to verify which operators obtain the best results for this problem.

Keywords: Forest Fires, Single-objective Optimization, Dispatching Problem, Genetic Algorithm

1 Introduction

Over the years, the number of forest fires on our planet has been very high, proving to be quite worrying, as they represent high-risk situations for the health of living beings and forests. Wildfires damage wildlife and the atmosphere, bringing serious environmental problems. In recent years, the causes of fires have been due to climate change, such as the increase in ambient temperature and due to the criminal hand. For example, between 2019 and early 2020, Australia was devastated by several forest fires caused by unprecedented high temperatures [6].

Every year, around 4 million square kilometers of land (roughly the size of the European Union) are burned worldwide [17]. In Europe, more than five thousand five hundred square kilometers of land burned in 2021, with one thousand square kilometers belonging to protected areas in the European Union [18]. In 2017,

* This work has been supported by FCT Fundação para a Ciência e Tecnologia within the R&D Units Project Scope UIDB/00319/2020 and PCIF/GRF/0141/2019: "O3F - An Optimization Framework to reduce Forest Fire" and the PhD grant reference UI/BD/150936/2021.

Portugal faced several forest fires, being a record year for the number of fires that occurred. The main causes were high temperatures and dry thunderstorms [7].

Firefighting is a very important and studied area in the literature. It is necessary to protect and avoid catastrophes, such as forest fires, and for this purpose studies are carried out to support the decisions of professionals in forest firefighting (firefighters, civil protection, etc.). The number, skills and level of preparation of firefighting teams are very important factors in fighting fires [1]. The better and faster the performance of the firefighting team, the less damage will be caused. The gradual increase in forest fires has been a worrying factor for society and for the planet Earth, since they cause deaths, pollution, damage to infrastructures, among other negative aspects. The management of the suppression of forest fires implies knowing how many and which combat resources must be dispatched for each forest fire, in order to extinguish the fire in the best way and quickly. For example, in the paper by Zeferino et al. [19] a mathematical optimization model was developed to find the best location solution for different aircraft that maximizes the coverage of risk areas. Its application was used in a case study in Portugal. In engineering, some optimization strategies have been used to solve problems associated with fire suppression aiming to find the optimal trajectory and location of combat vehicles, obtaining the shortest route, determining the number and which resources to dispatch, among others. In the work by Chan et al. [8] the Firefly algorithm was proposed to dispatch a limited amount of drones and firefighters between several zones. The performance of Firefly algorithm was evaluated in a wide variety of configurations, showing that when a relatively small number of drones are used (for example, 10 - 20% of the total number of zones) the algorithm can reach up to 80 - 95% of the optimal performance in most cases. A mathematical model was formulated in [10] for firefighting and rescue service dispatch problem. This problem aims to determine the allocation of firefighters to vehicles and how the vehicles should be dispatched for an emergency. The model was solved exactly and heuristically using data from Skåne, in Sweden. The results showed that the exact solution method is very time consuming in some cases, and that, in most cases, the heuristic finds an optimal solution.

Other studies were developed to determine how many and which resources are available and where and when to act in a given forest fire. In order to reduce forest fires, it is necessary to contribute with studies and develop support systems for forest management, using modern techniques for monitoring, detection and control [1]. Planning how many and which forest firefighting resources are needed to extinguish a given fire is a very relevant area of study that can lead to damage reduction and support decision makers in combat actions. A resource dispatch problem is defined as a problem that simulates where, when and what resources will act on the ground. This problem when applied to forest firefighting, is based on knowing which means of combat should be sent for a given fire and when to send them [2]. The genetic algorithm (GA) is one of the most used metaheuristics in this type of problem. Several works have been presented, applying the genetic algorithm to forest firefighting associated problems. A problem using a resource for multiple simultaneous ignitions was introduced by [14]. The objective was to find the optimal sequence of actions in firefighting, minimizing the total fire damage in the burned areas. They proposed a stochastic formulation to solve the problem concluding that the approach was effective and efficient.

Baisravan et al. [13] proposed several decision support strategies to minimize the total burned area. For this, the GA was applied to find the best strategy in order to reach the objective, using a certain number of resources. The GA builds an optimal line of fire to reduce the total area burned and provides the attacking teams with suitable locations for the line of fire to be built before the fire escapes. A significant decrease in the damage caused was verified. Later, Baisravan et al. [12] presented a GA-based approach for the development of efficient strategies in fire building lines, using intelligent dispatch of resources to minimize total damage to wild areas caused by fires. The approach used a simulation optimization technique where the GA uses advanced wildfire propagation models based on Huygens principles to evaluate the cost index. Homogeneous and heterogeneous environmental conditions were considered with uncertainty in meteorological conditions. Monte-Carlo simulations were used to develop robust strategies under uncertain conditions. With this approach it was possible to verify its effectiveness in the dispatch of resources to combat complex forest fires in uncertain and dynamic conditions. The work developed by Matos et al. [16] aimed to find an optimal scheduling of a forest firefighting resource in the combat of multiple ignitions. The goal was to minimize the total burned area, using GA in a problem with 10 forest fire ignitions located.

This work aims to study a resource dispatch problem for forest firefighting. It is intended to assign 7 resources of forest firefighting to combat 20 ignitions of a forest fire. An adapted GA, implemented in *Python* language, was used to minimize the total burned area of the ignitions. Furthermore, several GA operators, crossover, mutation and selection were tested, and a statistical analysis was carried out to verify which operators to apply in order to obtain the best results in this problem.

This paper is organized as follows. The genetic algorithm is described in Sect. 2, which is used to solve the problem presented in Sect. 3. The experimental results are shown in Sect. 4, where the statistical analysis of the results is also presented. Finally, the conclusions of this study and future work are exposed in Sect. 5.

2 Genetic Algorithm

Genetic Algorithms are well-known and commonly used optimization algorithms, originally proposed by John Holland in 1975 [11]. The genetic algorithm is a stochastic global optimization algorithm inspired by the evolutionary theory of species, namely natural selection, survival of the fittest and the inheritance of traits from parents to offspring by reproduction. The main components of a GA are the chromosome population, the operators (selection, crossover and mutation), the fitness function and the algorithm termination [15].

The GA can be described by the following pseudo-code:

1. Randomly initialize the population of individuals (chromosomes).
2. Compute the fitness function of each individual in the population.
3. Select individuals based on their fitness to serve as parents to form a new population.
4. Perform crossover and mutation on parents to create offspring to form a new population.
5. Copy the best individual to the new population (elitism).
6. Repeat steps 2-5 until termination condition is satisfied.

Initialization

GA is particularly suitable for exploring the search space of a combinatorial optimization problem, for example through a permutation representation of the individuals in population [4]. In this paper, a permutation representation was adopted to represent a possible solution to the problem of dispatching forest firefighting resources. In this representation, each chromosome is a sequence of integer values that can only appear once, that is, each combat resource can fight a certain number of ignitions at different instants of time, following a certain order of priority, which represents each chromosome [3]. The number of elements in the chromosome is given by multiplying the number of resources by the number of ignitions. An example representation of the permutation with 3 resources (R_1, R_2, R_3) and 4 ignitions (I_1, I_2, I_3, I_4), for each instant of time t, is presented in Table 1, where the chromosome length is 12. The permutation representation indicates the order of action of each resource, where the first four elements of the chromosome show the order of action of resource R_1, then from the 5th to the 8th element gives the order of action of resource R_2 and the last four elements give the order of action of resource R_3 in each ignition. Thus, at the initial instant of time, the chromosome in Table 1 indicates that the resource R_1 goes to ignition I_2, the resource R_2 goes to ignition I_1 and the resource R_3 goes to ignition I_3. Afterwards, R_1 goes to ignition I_3, the resource R_2 goes to ignition I_2 and the resource R_3 goes to ignition I_1 and so on.

Table 1. Permutation representation (example)

Resource	**R1**				**R2**				**R3**			
Chromossome	4	5	2	8	1	9	10	11	7	3	12	6
Ignitions Order	I_2	I_3	I_1	I_4	I_1	I_2	I_3	I_4	I_3	I_1	I_4	I_2

Fitness function

The fitness function measures the quality of the chromosome (in terms of solution) and is related to the objective function. In each generation, the fittest chromosomes in the population are more likely to be selected to generate offspring through crossover and mutation genetic operators.

Operators

The genetic material from the chromosomes is combined to ensure that promising new regions of the search space are explored. However, genetic operators must ensure that feasible permutations are maintained during the search. Several specialized genetic operators have been developed to meet this requirement [4]. Thus, the genetic operators that will be explored and tested for the resource dispatch problem for forest firefighting are shown in Table 2.

Table 2. GA operators explored this work

Crossover	**Mutation**	**Selection**
Edge recombination (ER)	Inverse (IM)	Tournament (TS)
Exponential (ExpC)	Bitflip (BM)	Random (RS)
Order-based (OR)	Polynomial (PM)	
Simulated Binary (SBX)		
Uniform (UC)		

Edge recombination crossover is a permutation (ordered) chromosome crossover operator aiming to introduce as few paths as possible between the various elements. In other words, edge recombination uses an adjacency matrix where there is a list of neighbors of each element of the parent chromosomes (see example in Fig. 1).

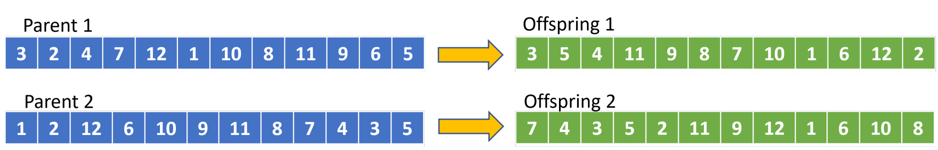

Fig. 1. Edge recombination crossover

The exponential crossover is similar to the one-point crossover or the two-points crossover. First, a chromosome position is chosen at random. Then a given number of consecutive positions are swapped between parents according to a decreasing exponential distribution [20], as can be seen in Fig. 2.

The order-based crossover is an operator used for permutation representations with the intention of transmitting information about the relative ordering to the offprints. An example of application of the order-based crossover can be visualized in Fig. 3.

The simulated binary crossover operator uses two parent vectors that give rise to two offspring solutions (see Fig. 4). This involves a parameter, called a distribution index that is held fixed in a positive value throughout a simulation.

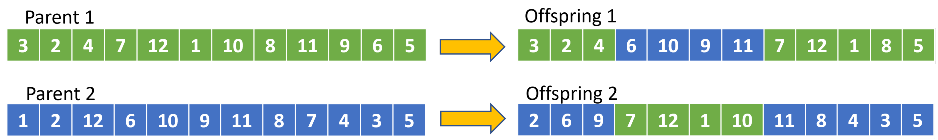

Fig. 2. Exponential crossover

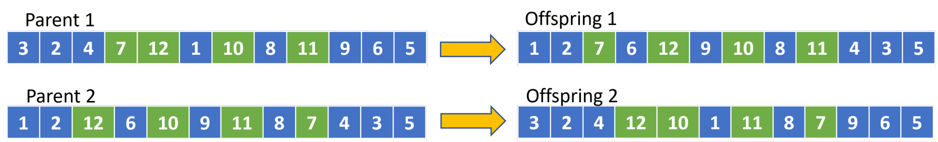

Fig. 3. Order-based crossover

If the distribution index value is large, then the resulting top-down solutions are close to the parent solutions. On the other hand, if the value of the distribution index is a small value, it is likely that the solutions are far from the parents [9].

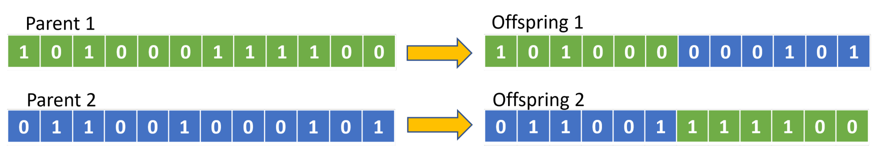

Fig. 4. Simulated binary crossover

The uniform crossover treats each gene on the chromosome individually, as can be seen in Fig. 5. In other words, you basically toss a coin to each gene and decide whether or not it will be included in the offspring.

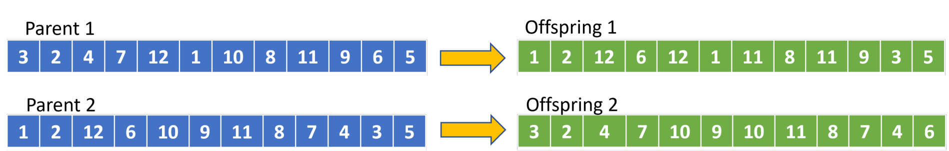

Fig. 5. Uniform crossover

The inversion mutation randomly selects some positions within a chromosome and inverts the genes on the subchromosome between those positions (see an example in Fig. 6).

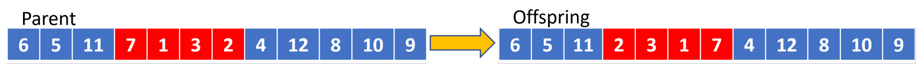

Fig. 6. Inversion mutation

The bitflip mutation operator randomly selects a gene from the parent chromosome, with binary encoding, and transforms it from 0 to 1 or vice versa, as shown in Fig. 7.

Fig. 7. Bitflip mutation

The polynomial mutation is similar to what occurs in SBX (Fig. 8), that is, a gene from the parent chromosome is selected and this is transformed into a random value (among the maximum number of genes on the chromosome) in the child chromosome.

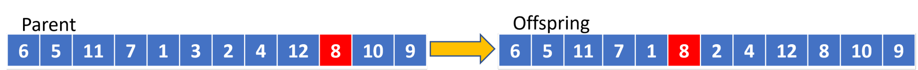

Fig. 8. Polynomial mutation

In random selection, only two chromosomes (parents) are selected to participate in mating and they cannot mate more than once, giving rise to offspring. So, successively, pairs of parents are selected at random without reposition from the population to generate offspring by the application of crossover and mutation.

Tournament selection, successively, selects at random two or more parent chromosomes and the one with the highest fitness function value is selected to generate offspring.

Termination

Termination is the final step of a GA, where the algorithm ends if it reaches some defined stopping criterion close to the optimal solution. For example, it ends when there are no improvements in the solution (the value of the objective function stagnates), when the maximum number of iterations is reached or when the objective function value reaches a certain predefined value. If the algorithm does not end, a new generation is performed where the GA operators (selection, crossover and mutation) are used to generate a new chromosome. This cycle is repeated until a certain stopping criterion is satisfied.

3 Problem Description

This work presents a resource dispatch problem for forest firefighting aiming to know which means of combat and when to send them to fight a fire. The objective is to minimize the total burned area, assigning 7 combat resources to suppress 20 ignitions of a wildfire. Thus, the goal is to determine, for each instant of time, which combat resource should go to each ignition, reducing the damage caused (total burned area).

For solving this problem, some assumptions are considered.

Ignition assumptions:

- Each ignition can only be extinguished once.
- Each ignition can be extinguished by one or more resources.
- The water required to extinguish a given ignition varies over time.
- For each ignition, the fire spreads and therefore the burned area varies over time.
- The distance between the base and each ignition, and between ignitions is known.
- Whenever the water capacity of the resources assigned to a given ignition is sufficient to extinguish it, it is extinguished instantly.

Resource assumptions:

- All resources, in the initial instant of time, are located in the base.
- Each resource has a maximum tank water capacity, that cannot be refilled when it is empty.
- At the initial instant of time, all resources have full tank water capacity.
- Each resource can only be assigned to one ignition at each instant of time.
- The resources velocity is considered constant.
- The travel time between the base and each ignition and between ignitions is the same for all resources.

The resource dispatch problem for forest firefighting is described as follows. The goal of this problem is to minimize the total burned area (TBA), using 7 resources (A, B, C, D, E, F and G) to extinguish 20 ignitions (I_i, $i = 1, \ldots, 20$) of a fire considering several instants of time. The data used in this work was generated to simulate a real situation. The resources are vehicles consisting of

Table 3. Resource capacity

Resource	**A**	**B**	**C**	**D**	**E**	**F**	**G**
Capacity	500	1000	3000	1500	1000	500	1500

water tanks totaling 9000 l of available water, whose capacity (in liters (l)) of each resource is listed in Table 3.

Table 4 shows the travel times (in minutes) between the location of each ignition and travel time from the base (Base) to each ignition. That is, it corresponds to the travel time from location where the 7 firefighting means are located at the beginning (in the Base), and each of the 20 ignitions and between each ignition (I_i). Travel times range between 10 and 100 minutes, with intervals of every 10 minutes $t = \{10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$.

Table 4. Travel time between ignition locations

I_i	**Base**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
Base	0	20	20	20	20	20	20	10	20	10	20	30	30	30	20	20	30	20	30	30	20
1	20	0	20	10	20	20	10	10	10	30	30	20	20	20	30	20	20	10	30	30	30
2	20	20	0	20	10	20	20	20	10	20	10	20	30	10	10	10	30	30	30	20	30
3	20	10	20	0	20	30	30	20	30	30	20	10	20	10	20	30	20	20	30	30	20
4	20	20	10	20	0	20	30	10	20	20	10	20	30	30	30	30	30	20	20	10	20
5	20	20	20	30	20	0	20	10	10	30	20	20	10	10	30	20	20	20	10	10	20
6	20	10	20	30	30	20	0	20	20	10	30	30	10	20	20	20	20	30	10	30	10
7	10	10	20	20	10	10	20	0	20	20	20	30	10	30	20	30	10	30	20	10	20
8	20	10	10	30	20	10	20	20	0	10	10	30	20	20	10	20	20	30	20	20	20
9	10	30	20	30	20	30	10	20	10	0	10	20	20	20	30	20	30	20	20	20	30
10	20	30	10	20	10	20	30	20	10	10	0	10	30	20	20	10	10	20	20	10	10
11	30	20	20	10	20	20	30	30	30	20	10	0	20	20	20	30	10	10	20	10	20
12	30	20	30	20	30	10	10	10	20	20	30	20	0	10	20	20	20	10	10	20	20
13	30	20	10	10	30	10	20	30	20	20	20	20	10	0	10	30	10	20	20	30	20
14	20	30	10	20	30	30	20	20	10	30	20	20	20	10	0	10	20	10	30	20	30
15	20	20	10	30	30	20	20	30	20	20	10	30	20	30	10	0	30	30	10	10	30
16	30	20	30	20	30	20	20	10	20	30	10	10	20	10	20	30	0	10	20	10	30
17	20	10	30	20	20	20	30	30	30	20	20	10	10	20	10	30	10	0	10	20	20
18	30	30	30	30	20	10	10	20	20	20	20	20	10	20	30	10	20	10	0	20	30
19	30	30	20	30	10	10	30	10	20	20	10	10	20	30	20	10	10	20	20	0	20
20	20	30	30	20	20	20	10	20	20	30	10	20	20	20	30	30	30	20	30	20	0

The burned area (in ha) of each ignition for each instant of time is shown in Table 5. The rows refer to each ignition I_i and the columns correspond to the instant of time t (in minutes). In each ignition, over time, the burned area increases, and for some ignitions the growth rate is lower than in others.

The amount of water (in liters) required to extinguish each ignition I_i at each instant of time t is shown in Table 6. For each instant of time, the larger

Table 5. Burned area for each ignition i at each instant of time t

I_i \ t	**10**	**20**	**30**	**40**	**50**	**60**	**70**	**80**	**90**	**100**
1	5.0	7.4	10.1	13.1	16.4	20.1	24.1	28.5	33.5	38.9
2	10.0	12.4	15.1	18.1	21.4	25.1	29.1	33.5	38.5	43.9
3	7.0	8.4	10.6	14.3	20.4	30.4	47.0	74.3	119.3	193.5
4	11.0	14.9	20.3	27.9	38.5	53.3	73.9	102.7	142.9	198.9
5	9.0	11.7	16.2	23.6	35.8	55.9	89.0	143.6	233.6	382.0
6	60.0	62.7	67.2	74.6	86.8	106.9	140.0	194.6	284.6	433.0
7	20.0	27.5	36.6	47.7	61.3	77.9	98.2	122.9	153.2	190.1
8	50.0	57.5	66.6	77.7	91.3	107.9	128.2	152.9	183.2	220.1
9	80.0	82.7	87.2	94.6	106.8	126.9	160.0	214.6	304.6	453.0
10	100.0	102.2	105.5	110.5	117.9	128.9	145.4	169.9	206.5	261.1
11	46.0	53.5	62.6	73.7	87.3	103.9	124.2	148.9	179.2	216.1
12	20.0	20.4	21.1	22.6	25.4	30.8	41.5	62.2	102.5	181.1
13	40.0	41.2	42.6	44.5	47.0	50.1	54.1	59.3	66.0	74.5
14	30.0	32.2	35.5	40.5	47.9	58.9	75.4	99.9	136.5	191.1
15	180.0	187.0	195.2	205.0	216.5	230.1	246.1	265.1	287.5	314.0
16	170.0	172.7	177.2	184.6	196.8	216.9	250.0	304.6	394.6	543.0
17	40.0	43.1	47.5	53.5	62.0	73.8	90.3	113.4	145.5	190.3
18	90.0	92.3	95.6	100.2	106.5	115.4	127.7	145.0	169.1	202.8
19	35.0	38.3	43.5	51.5	63.8	82.8	112.2	157.5	227.6	335.8
20	50.0	52.2	55.3	60.0	66.8	76.9	91.6	113.3	145.2	192.0

the burned area, the greater the amount of water needed to extinguish a given ignition.

4 Experimental Results

In this work, GA is implemented in the *Python* language, after being adapted from the *pymoo* framework: Single-objective Optimization in Python [5]. First, a permutation representation is used so that the solution to the problem takes the form of the order in which a sequence of events should occur, as described in Sect. 2. The permutations represent solutions of the problem, where an array with size equal to the number of resources times the number of ignitions was generated. Then the array is ranked in descending order of combat priority of ignitions for each resource. Finally, a reordering was applied to the resources that still have sufficient capacity to extinguish ignitions. Then, some GA operators are tested to determine the configuration that best performed in solving the problem of dispatching forest firefighting resources. Finally, a statistical analysis is carried out between the different tests of the GA operators (crossover, mutation and selection) to support the decision of which are the best operators to use to obtain the best solution for this problem. At the end, a discussion will be held on the results obtained.

Regarding the parameters used by GA, the population size was set to 20, and the GA default values for the maximum number of generations and the

Table 6. Water required for extinguish each ignition I_i at each instant of time t

I_i \ t	**10**	**20**	**30**	**40**	**50**	**60**	**70**	**80**	**90**	**100**
1	79.3	96.7	112.9	128.4	143.7	158.8	174.0	189.4	205.1	221.1
2	112.1	125.0	137.9	150.9	164.1	177.5	191.2	205.3	219.9	234.9
3	93.8	102.5	115.4	134.0	160.1	195.5	243.0	305.5	387.2	493.1
4	117.6	136.8	159.8	187.3	220.0	258.8	304.8	359.2	423.7	500.0
5	106.3	121.3	142.7	172.2	212.0	264.9	334.4	424.8	541.8	692.8
6	274.6	280.7	290.6	306.2	330.2	366.4	419.4	494.5	598.0	737.6
7	158.5	185.8	214.4	244.8	277.5	312.9	351.2	393.0	438.7	488.8
8	250.7	268.7	289.2	312.5	338.7	368.2	401.3	438.4	479.8	525.9
9	317.1	322.4	331.0	344.8	366.3	399.3	448.4	519.3	618.7	754.5
10	354.5	358.4	364.2	372.6	384.9	402.5	427.4	462.0	509.4	572.8
11	240.4	259.2	280.4	304.3	331.2	361.3	395.0	432.6	474.5	521.1
12	158.5	160.0	162.9	168.4	178.5	196.8	228.2	279.5	358.9	477.0
13	224.2	227.4	231.5	236.6	243.0	251.0	260.9	273.0	287.9	306.0
14	194.2	201.2	211.3	225.6	245.3	272.1	307.7	354.3	414.1	490.0
15	475.6	484.7	495.3	507.5	521.6	537.7	556.1	577.2	601.1	628.1
16	462.2	465.9	471.9	481.6	497.3	522.0	560.5	618.7	704.2	826.0
17	224.2	232.8	244.2	259.4	279.1	304.6	336.9	377.4	427.6	489.1
18	336.3	340.6	346.6	354.8	365.8	380.8	400.7	426.9	461.0	504.8
19	209.7	219.5	233.8	254.3	283.1	322.6	375.4	444.9	534.8	649.6
20	250.7	256.0	263.7	274.6	289.8	310.8	339.4	377.4	427.2	491.2

maximum number of function evaluations were 1000 and 100000, respectively. As GA is a stochastic algorithm, 30 independent runs were performed in order to statistically analyze its performance.

The numerical experiments were carried out on a PC 11th Gen Intel(R) Core(TM) i7-1165G7 @ 2.80GHz, 2803 Mhz, 4 Nucleus(s), 8 Processor(s) Logic(s), 16 Gb RAM. The code was implemented in *python* (version 3.9.13) using *VScode* (Version 1.77).

4.1 Testing GA Operators

The strategy used for testing the GA operators presented in Table 2 was as follows. First, keeping the default selection and mutation operators of the *pymoo* framework (Tournament Selection and Inverse Mutation), the crossover operator was varied. Then, for the crossover operator that obtained the best performance in terms of total burned area, the Tournament Selection was maintained and the mutation operator was varied. Finally, for the crossover and mutation operator that obtained the best result, the other selection operator (Random Selection) was tested.

Table 7 shows the average solution values, among the 30 runs, for the total burned area (TBA$_{av}$), the number of objective function evaluations (nfe$_{av}$), the total remaining water (RW$_{av}$), the total water used (UW$_{av}$) and the execution time (Time$_{av}$), in seconds. In addition, the standard deviation (St dev) is also reported. The solution with the smallest average value of the total burned area found is marked in bold.

Table 7. Average solution values for different crossover, mutation and selection operators

Selection	Crossover	Mutation	$\mathbf{TBA_{av}}$	$\mathbf{RW_{av}}$	$\mathbf{UW_{av}}$	St dev	$\mathbf{nfe_{av}}$	$\mathbf{Time_{av}}$
TS	ER	IM	1322.1	2781.2	6218.8	26.8	620.0	11.1
TS	ExpC	IM	1277.4	2847.9	6152.1	27.1	620.0	6.7
TS	OR	IM	1306.8	2718.6	6281.4	24.1	620.0	5.4
TS	SBX	IM	1306.5	2786.7	6213.3	30.0	620.0	4.1
TS	UC	IM	1303.9	2992.0	6008.0	27.1	620.0	3.6
TS	ExpC	BM	1272.4	3065.0	5935.0	35.4	628.7	5.3
TS	**ExpC**	**PM**	**1249.9**	**3108.8**	**5891.2**	**36.2**	**1584.7**	**16.1**
RS	ExpC	PM	1251.8	3027.4	5972.6	30.6	1540.7	21.1

In a first approach, a statistical analysis was performed among the GA crossover operators. Thus, the crossover operator, with the lowest value of the total burned area, was chosen (Exponential, ExpC). Then, the GA mutation operators IM and PM were tested. Finally, the best mutation operator was chosen (Polynomial, PM), and GA selection operator RM was also tested. For the statistical analysis, a one-tailed paired sample t-student test was used, where the p-values and the differences between the TBA values of each operator are presented in the next tables. In all tests, a significance level of 5% was considered.

Testing Crossover

As mentioned earlier, the several GA crossover operators were tested by setting the default operators of *pymoo* for selection and mutation (TS and IM). Table 8 shows the p-values, in the lower triangular part of the table, and average differences between the TBA values of the crossover operators, in the upper triangular part of the table.

Table 8. Statistical analysis for crossover operator

	ExpC	ER	OR	SBX	UC
ExpC	-	**-44.789**	**-29.446**	**-29.169**	**-26.561**
ER	<0.001	-	**15.343**	**15.620**	**18.228**
OR	<0.001	0.012	-	0.277	2.886
SBX	<0.001	0.019	0.484	-	2.608
UC	<0.001	0.006	0.332	0.363	-

With the crossover operator ExpC, the best result was obtained, when compared to all other crossover operators, since the TBA values were negative. On the other hand, the ER was the worst operator compared to the others (OR, SBX and UC crossovers), where the difference between the TBA values of the various operators was positive. The statistical analysis showed that there were significant differences between the crossover operator ExpC and the others, since the p-value was less than 0.05. When comparing the p-values of the ER crossover with those OR, SBX and UC, the statistical analysis indicates that the differences are significant, since p-values are less than 0.05. Finally, comparing the crossovers OR with SBX, OR with UC and SBX with UC, it is possible to notice that there were no significant differences between these operators, as the p-values were greater than 0.05. Therefore, the best operator was ExpC.

Testing Mutation

Then, the previously chosen crossover operator (ExpC) was fixed and the TS was maintained, a statistical analysis was performed between the GA mutation operators. As can be seen in Table 9 there were no significant differences between IM and BM operators, as the p-value is greater than the 0.05. In addition, it was also possible to observe that the PM operator stood out from the IM and BM operators, since the p-value was less than 0.05, showing significant differences between them. Thus, it was possible to conclude that the best GA mutation operator was the PM.

Table 9. Statistical analysis for mutation operator

	IM	**BM**	**PM**
IM	-	4.962	**27.413**
BM	0.272	-	**22.451**
PM	<0.001	0.009	-

Testing Selection

Finally, after the best crossover and mutation operators had been previously chosen (ExpC and PM), a statistical analysis was performed between the RS and TS operators. In Table 10, it can be seen that there were no significant differences (p-value greater than 0.05), which means that it is indifferent to use RS or TS.

Table 10. Statistical analysis for selection operator

	RS	**TS**
RS	-	1.905
TS	0.413	-

After the statistical analysis, it was found that the best operators for this problem were ExpC and PM. Concerning the selection operator, the TS operator was selected due to the lowest average value of TBA (marked in bold in Table 7).

4.2 Best Result Analysis

The best solution found by GA, among the 30 runs, when using the best operators previously chosen, ExpC, PM and TS, is presented in Table 11. It shows the optimal value obtained for TBA, the number of function evaluations (nfe) and the execution time (Time). The table also shows the values of the remaining water (RW) and the used water (UW) in that best solution.

Table 11. Best solution found using ExpC, PM and TS operators

	TBA	**RW**	**UW**	**nfe**	**Time**
Best Solution	1236.2	3976.0	5024.0	1900.0	19.5

Table 12 shows the best solution found by GA in terms of the instant of time each resource is assigned to each ignition. The symbol $\rightarrow$ represents that the resource of combat is traveling.

Table 12. Best solution found by GA for TBA = 1236.2 ha

R \ t		**10**	**20**	**30**	**40**	**50**
A	Base	$\rightarrow$	I_4	I_{19}		
B	Base	$\rightarrow$	I_3	I_{11}	$\rightarrow$	I_{20}
C	Base	I_9	$\rightarrow$	I_{15}	I_2	I_{13}
D	Base	I_7	I_1	I_6	I_{18}	I_{12}
E	Base	$\rightarrow$	I_{16}	I_5	I_{10}	
F	Base	$\rightarrow$	I_8	$\rightarrow$	I_{20}	
G	Base	$\rightarrow$	I_{14}	I_{17}		

In the beginning, all the resources are at the Base. At $t = 10$, resources C and D are assigned to ignitions I_9 and I_7, respectively. At $t = 20$ resources A, B, D, E, F and G are assigned to extinguish ignitions I_4, I_3, I_1, I_{16}, I_8 and I_{14}, respectively. In addition, resource C is assigned to ignition I_{15}, but the trip from I_9 to I_{15} takes 20 minutes (see Table 4), so at $t = 20$ it is traveling. The ignitions I_{19}, I_{11}, I_{15}, I_6, I_5 and I_{17} are extinguished in the instant of time $t = 30$. At this instant of time, resource A is not assigned to any further ignitions, as it does not have enough water capacity to extinguish any still active ignition (remain water of resource A is 129.4 l). At the instant of time $t = 40$, resources C, D and E are dispatched to ignitions I_2, I_{18} and I_{10}, respectively, extinguishing them. Resource F is assigned to ignition I_{20}, but does not have enough water to extinguish this ignition by itself, running out of water. At this time ($t = 40$) the ignition I_{20} requires 274.6 l of water (see Table 6), but resource F only has 231.3 l of water. Since resource F cannot extinguish I_{20} alone, resource B is assigned to support extinguishing this ignition at time $t = 50$. At this time, I_{13} and I_{12} are extinguished by resources C and D, respectively. Note that resource F has used all of its water tank capacity. Although some resources were low on water, or even without water, a total of 3976.0 l of water still remained. Thus, at time $t = 50$ all ignitions were extinguished and the total water used was 5024.0 l.

5 Conclusions and Future Work

The occurrence of forest fires has increased in recent years and is essentially due to natural or human factors. Therefore, it is necessary to look for solutions that can manage fire suppression, such as optimizing firefighting actions.

In this work, a resource dispatch problem for forest firefighting was addressed. The problem was based on 7 resources that would have to be assigned to 20 ignitions in order to extinguish them, minimizing the total burned area. For this, the metaheuristic GA from the *pymoo* framework was used, and adapted with permutation representation to obtain the optimal solution of the problem. Furthermore, several GA operators, crossover, mutation and selection were tested, and a statistical analysis was carried out to verify which operators to apply in order to obtain the best results in this problem. After this analysis, it was found that the best operators were ExpC crossover, PM mutation and TS selection. Then, the optimal solution found in terms of total burned area was TBA = 1236.2 ha using the best GA operators. By analyzing this solution, it was possible to identify at what instant of time each resource goes to each ignition. With this approach it was possible to realize that the strategy was effective and fast.

In the future, it is intended to deal with this problem but using real data. A multi-objective approach can also be applied to the resource dispatch problem for forest firefighting, minimizing simultaneously the total burned area and the used water.

References

1. Attri, V., Dhiman, R., Sarvade, S.: A review on status, implications and recent trends of forest fire management. Archives of Agriculture and Environmental Science **5**(4), 592–602 (2020)
2. Bélanger, V., Lanzarone, E., Nicoletta, V., Ruiz, A., Soriano, P.: A recursive simulation-optimization framework for the ambulance location and dispatching problem. European Journal of Operational Research **286**(2), 713–725 (2020)
3. Bessedik, M., Benbouzid-Si Tayeb, F., Cheurfi, H., Blizak, A.: An immunity-based hybrid genetic algorithms for permutation flowshop scheduling problems. The International Journal of Advanced Manufacturing Technology **85**, 2459–2469 (2016)
4. Bierwirth, C., Mattfeld, D.C., Kopfer, H.: On permutation representations for scheduling problems. In: International Conference on Parallel Problem Solving from Nature. pp. 310–318. Springer (1996)

5. Blank, J., Deb, K.: pymoo: Multi-objective optimization in python. IEEE Access **8**, 89497–89509 (2020)
6. Boer, M.M., de Dios, V.R., Bradstock, R.A.: Unprecedented burn area of australian mega forest fires. Nature Climate Change **10**(3), 171–172 (2020)
7. Carmo, I.I.V.: O papel dos Instrumentos de Gestão Territorial na prevenção e mitigação dos incêndios florestais: o caso do incêndio de Pedrogão Grande (2017) (in Portuguese). Ph.D. thesis (2021)
8. Chan, H., Tran-Thanh, L., Viswanathan, V.: Fighting wildfires under uncertainty: A sequential resource allocation approach. In: Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence. pp. 4322–4329 (2021)
9. Deb, K., Sindhya, K., Okabe, T.: Self-adaptive simulated binary crossover for real-parameter optimization. In: Proceedings of the 9th annual conference on genetic and evolutionary computation. pp. 1187–1194 (2007)
10. Granberg, T.A.: Optimized dispatch of fire and rescue resources. In: Computational Logistics: 13th International Conference, ICCL 2022, Barcelona, Spain, September 21–23, 2022, Proceedings. pp. 132–146. Springer (2022)
11. Holland, J.H.: Adaptation in Natural and Artificial Systems. MIT Press, Cambridge, MA, USA (1975)
12. HomChaudhuri, B., Kumar, M., Cohen, K.: Genetic algorithm based simulation–optimization for fighting wildfires. International Journal of Computational Methods **10**(06), 1350035 (2013)
13. HomChaudhuri, B., Zhao, S., Cohen, K., Kumar, M.: Generation of optimal fire-line for fighting wildland fires using genetic algorithms. In: Dynamic Systems and Control Conference. vol. 48920, pp. 111–118 (2009)
14. Kali, A.: Stochastic scheduling of single forest firefighting processor. Canadian Journal of Forest Research **46**(3), 370–375 (2016)
15. Lambora, A., Gupta, K., Chopra, K.: Genetic algorithm- a literature review. In: 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon). pp. 380–384 (2019). https://doi.org/10.1109/COMITCon.2019.8862255
16. Matos, M.A., Rocha, A.M.A., Costa, L.A., Alvelos, F.: A genetic algorithm for forest firefighting optimization. In: Computational Science and Its Applications–ICCSA 2022 Workshops: Malaga, Spain, July 4–7, 2022, Proceedings, Part II. pp. 55–67. Springer (2022)
17. Naderpour, M., Rizeei, H.M., Khakzad, N., Pradhan, B.: Forest fire induced natech risk assessment: A survey of geospatial technologies. Reliability Engineering & System Safety **191**, 106558 (2019)
18. San-Miguel-Ayanz, J., Durrant, T., Boca, R., Maianti, P., Liberta, G., Artes, T.V., Oom, D.J.F., Branco, A., De Rigo, D., Ferrari, D., Pfeiffer, H., Grecchi, R., Onida, M., Loffler, P.: Forest fires in europe, middle east and north africa 2021 (KJ-NA-31-269-EN-N (online), KJ-NA-31-269-EN-C (print)) (2022). https://doi.org/10.2760/34094
19. Zeferino, J.A.: Optimizing the location of aerial resources to combat wildfires: a case study of Portugal. Natural Hazards **100**(3), 1195–1213 (2020)
20. Zhao, S.Z., Suganthan, P.N.: Empirical investigations into the exponential crossover of differential evolutions. Swarm and Evolutionary Computation **9**, 27–36 (2013)