Auto-adaptation of Genetic Operators for Multi-objective Optimization in the Firefighter Problem

Krzysztof Michalak

Department of Information Technologies,
Institute of Business Informatics,
Wroclaw University of Economics, Wroclaw, Poland
krzysztof.michalak@ue.wroc.pl

Abstract. In the firefighter problem the spread of fire is modelled on an undirected graph. The goal is to find such an assignment of firefighters to the nodes of the graph that they save as large part of the graph as possible.
In this paper a multi-objective version of the firefighter problem is proposed and solved using an evolutionary algorithm. Two different auto-adaptation mechanisms are used for genetic operators selection and the effectiveness of various crossover and mutation operators is studied.

Keywords: operator auto-adaptation, multi-objective evolutionary optimization, graph-based optimization, firefighter problem

1 Introduction

The firefighter problem was introduced by Hartnell in 1995 [14]. It can be used as a deterministic, discrete-time model for studying the spread and containment of fire, containment of floods and the dynamics of the spread of diseases.

The problem definition is as follows. Let $G = \langle V, E \rangle$ be an undirected graph, and $L = \{'B', 'D', 'U'\}$ a set of labels that can be assigned to the vertices of the graph G. The meaning of the labels is $'B'$ = burning, $'D'$ = defended and $'U'$ = untouched. Let $l : V \to L$ be a function that labels the vertices. Initially, all the vertices in V are marked $'U'$ (untouched). At the initial time step $t = 0$ a fire breaks out at a non-empty set of vertices $\emptyset \neq S \subset V$. The vertices from the set S are labelled $'B'$: $\forall v \in S : l(v) =' B'$. At every following time step $t = 1, 2, \ldots$ a predefined number d of yet untouched nodes (labelled $'U'$) become defended by firefighters (these nodes are labelled $'D'$). A node, once marked $'D'$, remains protected until the end of the simulation. Each time step finishes with the fire spreading from the nodes labelled $'B'$ to all the neighbouring nodes labelled $'U'$. The simulation ends when either the fire is contained (i.e. there are no undefended nodes to which the fire can get) or when all the undefended nodes are burning. The goal is to find an assignment of firefighters to d nodes per each time step $t = 1, 2, \ldots$, such that, when the simulation stops, the number of saved vertices (labelled $'D'$ or $'U'$) is maximal.

In this paper a multi-objective version of the firefighter problem is tackled in which there are m values $v_i(v)$, $i = 1, \ldots, m$ assigned to each node v in the graph. Each v_i value can be interpreted as a worth of a node with respect to a different criterion (e.g. the financial worth vs. the cultural value). The objectives f_i, $i = 1, \ldots, m$ attained by a given solution are calculated as follows:

$$f_i = \sum_{v \in V : l(v) \neq' B'} v_i(v) \tag{1}$$

where:
$v_i(v)$ is the value of a given node according to the i-th criterion.

Many papers published to date on the firefighter problem deal with theoretical properties and concern specific types of graphs and specific problem cases. Obviously, such results are not applicable in the general case when specific assumptions may not be guaranteed to be true. In the paper [8] a linear integer programming model was proposed, however it was only a single-objective one. To date, few papers have been published on using metaheuristic methods for solving the firefighter problem. A recent paper [4] states even, that before its publication not a single metaheuristic approach has been applied to the firefighter problem. In the aforementioned paper an Ant Colony Optimization (ACO) approach was proposed for a single-objective case. In this paper a multi-objective evolutionary algorithm with operator auto-adaptation is used.

The rest of this paper is structured as follows. Section 2 presents the algorithm used for solving the multi-objective firefighter problem. In Section 3 the experimental setup is presented along with the obtained results. Section 4 concludes the paper.

2 Algorithm

Evolutionary algorithms are often used for multi-objective optimization. The advantage of this type of optimization methods is that they maintain an entire population of solutions which may represent various trade-offs between the objectives. Since the problem presented in this paper is multi-objective the NSGA-II algorithm [7] is used which is used in the literature in many areas including engineering applications [13] and operations research [18]. The optimization problem considered in this paper involves m objectives which represent the value of the graph nodes with respect to different criteria. In the algorithm an $m + 1$-th objective is added which represents the number of nodes saved (i.e. labelled either 'D' or 'U' at the end of the simulation). This objective is added in order to promote solutions that allow the fire to be contained early.

The genotype used in the algorithm is a permutation P of N_v elements. This permutation represents the order in which nodes of the graph are defended by firefighters. During the simulation of the spreading of fire firefighters are assigned to d nodes of the graph at the time in the order determined by the permutation P. If a given node is already labelled $'B'$ then a firefighter is assigned to the next untouched node from the permutation P. At a given time step d firefighters are always assigned, even if some nodes are skipped because they are already marked $'B'$.

Genetic operators: A set of 10 crossover operators and 5 mutation operators is used for genetic operations. The crossover operators are: Cycle Crossover (CX) [17], Linear Order Crossover (LOX) [9], Merging Crossover (MOX) [1, 16], Non-Wrapping Order Crossover (NWOX) [6], Order Based Crossover (OBX) [19], Order Crossover (OX) [11], Position Based Crossover (PBX) [19], Partially Mapped Crossover (PMX) [12], Precedence Preservative Crossover (PPX) [3, 2] and Uniform Partially Mapped Crossover (UPMX) [5]. The mutation operators are: displacement mutation, insertion mutation, inversion mutation, scramble mutation and transpose mutation.

Auto-adaptation mechanism: The effectiveness of genetic operators may vary for different problems, different instances of a given problem and even may change at different phases of the optimization process. In order to choose the best performing crossover and mutation operators an auto-adaptation mechanism can be used [15]. In this paper two auto-adaptation mechanisms are compared. The first mechanism (RawScore) uses a raw score calculated as the number of times b_i when a given operator produced an improved specimen. Note, that in the case of the crossover operator each offspring is compared to each parent, so if one offspring is better than both parents then $b_i = 2$ and if both offspring are better than both parents then $b_i = 4$. Also, in the case of a multi-objective problem an improvement along any of the objectives is counted.

The second mechanism (SuccessRate) is based on the success rate of the operators. This mechanism counts the number of times each operator was used n_i and the number of improvements obtained b_i. Similarly as with the first mechanism each offspring is compared to each parent, so a maximum value of $b_i = 4$ can be obtained (per objective). The success rate is calculated as $s_i = b_i/n_i$ if $n_i \neq 0$ or $s_i = 0$ if $n_i = 0$.

Each of the N_{op} operators is given a minimum probability P_{min} and the remaining $1 - N_{op}P_{min}$ is divided proportionally to either the raw scores b_i (in the RawScore method) or the success rate values s_i (in the SuccessRate method) obtained by the operators. In both methods operators are selected randomly using a roulette-wheel selection principle. The selection of crossover and mutation operators is performed separately with separate probability assignment procedures.

3 Experiments and Results

In the experiments the optimization of firefighter assignment was performed for graphs with various density of edges. The graphs were built as follows. First, a number N_v of vertices were created. Then, edges were added with a probability P_{edge} of creating an edge between any given two vertices. The density of the graph heavily impacts the progress of the simulation. If the density is low the fire is easily contained. If the density is high the fire is very hard to contain and all nodes burn except those protected by firefighters (no nodes with the label 'U' at the end of simulation). Therefore, the value of P_{edge} was selected based on a preliminary round of tests in order to ensure that on one hand some of the nodes are left in the untouched state 'U' and on the other hand the fire is not stopped immediately. The number of vertices N_s at which the fire started was set to 1, $0.04 \cdot N_v$ and $0.1 \cdot N_v$. The number of firefighters N_f assigned at each time step was set to $N_f = 2 \cdot N_s$. The parameters of the test instances are presented in Table 1.

Table 1. Parameters of the test instances

N_v	P_{edge}	N_s
50	0.05	1, 2 and 5
100	0.02	1, 4 and 10
500	0.0055	1, 20 and 50
1000	0.0025	1, 40 and 100

The parameters of the evolutionary algorithm were set as follows in the experiments. The number of generations was set to $N_{gen} = 250$. In order to allow larger populations for larger problem instances the population size was set to be equal to the number of the nodes in the graph $N_{pop} = N_v$. Specimens for a new generation were generated by applying a crossover operator and then mutation operator. The number of new specimens generated by the crossover operator was equal to the population size N_{pop} and the probability of mutation was set to $P_{mut} = 0.05$. The minimum probability of selecting a particular operator in the auto-adaptation mechanism was set to $P_{min} = 0.02$ for crossover auto-adaptation and to $P_{min} = 0.05$ for mutation auto-adaptation.

The auto-adaptation mechanism changes the probabilities with which individual operators are selected. At various stages of the optimization process different operators may be used more often than others. An example of this effect is visible in Figure 1 which presents the success rates of crossover operators plotted against the generation number for the instance with $N_v = 1000$ vertices and $N_s = 100$ fire starting points. Clearly, around generation 40 the MOX crossover works best followed closely by the PPX. Near the end of the search the OBX and PBX crossovers work best.

The results of multi-objective optimization performed using different algorithms are often hard to compare because individual solutions may dominate one another with respect to different objectives. In order to obtain numeric values that represent the quality of the generated results various measures of the Pareto front quality are used. Hypervolume indicator introduced in [20] is very often used for that purpose. It has been proven in [10] that maximizing the hypervolume is equivalent to achieving Pareto optimality. Table 2 presents the hypervolume values obtained in the experiments for both auto-adaptation methods. The presented values are averages calculated from 10 runs of the test performed for each set of parameters.

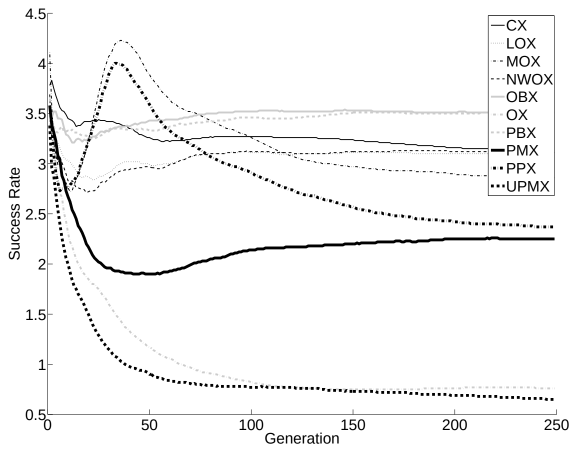

Fig. 1. The success rates of crossover operators plotted against the generation number for the instance with $N_v = 1000$ vertices and $N_s = 100$ fire starting points

Table 2. Hypervolume values attained by both auto-adaptation methods

N_v	P_{edge}	N_s	RawScore auto-adaptation	SuccessRate auto-adaptation
50	0.05	1	$4.6143 \cdot 10^6$	$\underline{5.2143 \cdot 10^6}$
		2	$5.5699 \cdot 10^7$	$\underline{5.6312 \cdot 10^7}$
		5	$5.2817 \cdot 10^7$	$\underline{5.4063 \cdot 10^7}$
100	0.02	1	$4.6310 \cdot 10^8$	$\underline{7.2057 \cdot 10^8}$
		4	$4.5708 \cdot 10^8$	$\underline{5.2746 \cdot 10^8}$
		10	$3.9548 \cdot 10^8$	$\underline{5.1009 \cdot 10^8}$
500	0.0055	1	$2.0798 \cdot 10^7$	$\underline{2.1406 \cdot 10^7}$
		20	$3.3457 \cdot 10^9$	$\underline{3.5775 \cdot 10^9}$
		50	$\underline{1.1501 \cdot 10^{10}}$	$0.9089 \cdot 10^{10}$
1000	0.0025	1	$\underline{1.0167 \cdot 10^8}$	$1.0016 \cdot 10^8$
		40	$\underline{2.7972 \cdot 10^{10}}$	$2.6013 \cdot 10^{10}$
		100	$\underline{1.1303 \cdot 10^{11}}$	$1.0949 \cdot 10^{11}$

In the experiments using the auto-adaptation based on the raw score the scores b_i were recorded for each operator $i = 1, \ldots, N_{op}$ separately for crossover and mutation operators. Recorded scores are presented in Tables 3 and 4. Among the crossover operators the CX crossover seems to work best as it has obtained the highest score most often. However, other crossover operators also produced good results on some instances. In the case of mutation operators the insertion mutation clearly performs best.

In the experiments using the auto-adaptation based on the success rate the success rates s_i were recorded for each operator $i = 1, \ldots, N_{op}$ separately for crossover and mutation operators. Recorded success rates are presented in Tables 5 and 6. Similarly as with the first auto-adaptation method the CX crossover achieves high success rates on some instances. Also, the OBX and PBX crossover operators seem to work well. Again, the insertion mutation achieves the best success rate outperforming, on average, all the other mutation operators.

Table 3. Scores obtained by crossover operators in the case of the auto-adaptation method based on raw scores

N_v	P_{edge}	N_s	CX	LOX	MOX	NWOX	OBX	OX	PBX	PMX	PPX	UPMX
		1	$\underline{339.7}$	172.9	312.5	210.7	272.7	148.0	182.6	167.9	140.4	231.3
50	0.05	2	$\underline{703.9}$	232.9	159.5	252.4	210.5	121.5	126.7	340.6	335.0	326.6
		5	197.9	120.6	$\underline{598.7}$	227.2	215.6	270.1	205.8	112.2	149.5	93.6
		1	$\underline{770.7}$	314.1	269.7	532.3	248.8	106.4	609.1	475.4	447.0	258.8
100	0.02	4	1112.6	949.7	1107.3	$\underline{1440.8}$	1160.9	458.6	871.5	801.8	643.8	449.1
		10	917.8	$\underline{1361.1}$	605.0	828.8	620.6	263.2	1084.6	1267.2	847.1	392.4
		1	4905.4	2907.4	4662.4	3410.5	6970.6	1293.1	8531.8	3733.0	$\underline{8706.2}$	4891.3
500	0.0055	20	20910.5	12966.6	25450.5	15624.4	17796.1	1330.2	$\underline{29291.9}$	7203.9	24431.8	2608.8
		50	$\underline{28899.5}$	15642.5	14174.3	15757.6	26394.8	2181.3	22251.9	6649.2	11675.2	1408.3
		1	12264.4	7023.0	9990.5	5643.3	17829.5	2150.1	17231.3	8418.1	$\underline{20872.6}$	8802.2
1000	0.0025	40	$\underline{60152.2}$	38021.3	36766.7	47273.8	44378.3	3968.4	47085.1	15657.3	58720.6	2654.3
		100	$\underline{65961.7}$	33573.6	37139.4	41497.6	59235.0	2671.5	61871.8	15297.6	33518.0	2224.6
AVERAGE			$\underline{16428.0}$	9440.5	10936.4	11058.3	14611.1	1246.9	15778.7	5010.4	13373.9	2028.4

Table 4. Scores obtained by mutation operators in the case of the auto-adaptation method based on raw scores

N_v	P_{edge}	N_s	displacement	insertion	inversion	scramble	transpose
		1	$\underline{16.6}$	8.7	8.1	8.4	2.3
50	0.05	2	6.3	$\underline{18.6}$	3.8	10.8	5.8
		5	$\underline{10.1}$	8.0	4.1	3.1	5.0
		1	16.3	6.1	$\underline{22.7}$	15.3	3.8
100	0.02	4	17.3	25.3	13.8	$\underline{26.6}$	9.9
		10	$\underline{26.5}$	23.7	12.3	$\underline{26.5}$	10.4
		1	$\underline{113.7}$	5.8	111.0	86.8	3.4
500	0.0055	20	121.1	$\underline{401.7}$	63.5	62.8	163.9
		50	80.3	$\underline{581.1}$	25.8	63.8	124.9
		1	132.1	6.6	113.8	$\underline{162.1}$	3.5
1000	0.0025	40	402.8	$\underline{1107.6}$	112.5	176.3	353.0
		100	243.2	$\underline{1165.5}$	53.0	138.7	339.6
AVERAGE			98.9	$\underline{279.9}$	45.4	65.1	85.5

Table 5. Success rates of the crossover operators generated by the auto-adaptation method based on success rates

N_v	P_{edge}	N_s	CX	LOX	MOX	NWOX	OBX	OX	PBX	PMX	PPX	UPMX
		1	0.356	0.329	0.346	0.341	0.411	0.251	$\underline{0.424}$	0.331	0.347	0.338
50	0.05	2	0.434	0.422	0.425	0.394	0.432	0.312	0.433	0.409	$\underline{0.456}$	0.359
		5	$\underline{0.395}$	0.330	0.336	0.343	0.356	0.280	0.357	0.318	0.331	0.278
		1	0.380	0.325	0.366	0.353	$\underline{0.396}$	0.190	0.386	0.292	0.390	0.323
100	0.02	4	$\underline{0.797}$	0.718	0.640	0.720	0.696	0.348	0.697	0.644	0.648	0.444
		10	$\underline{0.865}$	0.802	0.730	0.787	0.743	0.342	0.756	0.706	0.753	0.461
		1	0.795	0.713	0.839	0.713	$\underline{1.076}$	0.305	1.068	0.648	1.015	1.016
500	0.0055	20	2.601	2.532	2.575	2.540	2.670	0.612	$\underline{2.684}$	1.983	2.495	0.875
		50	2.563	2.611	2.644	2.606	2.865	0.563	$\underline{2.874}$	2.033	2.297	0.885
		1	0.803	0.671	0.798	0.676	0.990	0.361	0.999	0.574	$\underline{1.009}$	0.849
1000	0.0025	40	$\underline{3.149}$	2.924	2.957	2.929	3.142	0.665	3.146	2.101	3.130	0.693
		100	3.151	3.082	2.770	3.090	3.399	0.558	$\underline{3.407}$	2.140	2.299	0.520
AVERAGE			1.357	1.288	1.286	1.291	1.431	0.399	$\underline{1.436}$	1.015	1.264	0.587

Table 6. Success rates of the mutation operators generated by the auto-adaptation method based on success rates

N_v	P_{edge}	N_s	displacement	insertion	inversion	scramble	transpose
		1	$\underline{0.054}$	0.050	0.053	0.033	0.048
50	0.05	2	0.057	$\underline{0.086}$	0.031	0.039	0.058
		5	0.023	$\underline{0.050}$	0.021	0.023	0.049
		1	$\underline{0.056}$	0.036	0.035	0.041	0.022
100	0.02	4	0.051	$\underline{0.102}$	0.058	0.055	0.076
		10	0.068	$\underline{0.089}$	0.052	0.067	0.045
		1	$\underline{0.057}$	0.019	0.044	0.049	0.010
500	0.0055	20	0.097	$\underline{0.213}$	0.060	0.073	0.131
		50	0.074	$\underline{0.231}$	0.048	0.051	0.154
		1	$\underline{0.049}$	0.011	0.041	0.040	0.003
1000	0.0025	40	0.118	$\underline{0.275}$	0.078	0.094	0.186
		100	0.080	$\underline{0.263}$	0.043	0.055	0.168
AVERAGE			0.065	$\underline{0.119}$	0.047	0.052	0.079

4 Conclusion

In this paper two auto-adaptation mechanisms for operator selection are tested on the firefighter problem. One of the auto-adaptation mechanisms uses raw scores calculated as the number of times each operator produces an improved specimen. The second mechanism uses scores normalized by the number of tries (i.e. the number of times each operator was applied). The first mechanism seems to work better on larger instances, while the first one produces better results on smaller instances.

Auto-adaptation based on raw scores seems to favor the CX crossover, while the auto-adaptation based on success rates gives a bit higher priority to the OBX and PBX crossovers. In the case of mutation both methods give the highest scores to the insertion mutation operator. The average score and success rate is clearly the highest in the case of insertion mutation. The inversion, scramble and transpose mutation operators obtain very poor results and are rarely used.

The fact that success rates of crossover operators vary over the duration of the optimization process motivates using auto-adaptation mechanisms because the performance of any individual operator may deteriorate over certain periods of time.

References

1. Anderson, P. G., A.D.: Advances in ordered greed. In: Dagli, C.H. (ed.) Intelligent Engineering Systems through Artificial Neural Networks, Proceedings of ANNIE 2004 International Conference. pp. 223–228. ASME Press, New York (2004)
2. Bierwirth, C., Mattfeld, D.C., Kopfer, H.: On permutation representations for scheduling problems. In: In 4th PPSN. pp. 310–318. Springer-Verlag (1996)
3. Blanton, Jr., J.L., Wainwright, R.L.: Multiple vehicle routing with time and capacity constraints using genetic algorithms. In: Proceedings of the 5th International Conference on Genetic Algorithms. pp. 452–459. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1993)

4. Blum, C., Blesa, M.J., Garcia-Martinez, C., Rodriguez, F.J., Lozano, M.: The firefighter problem: Application of hybrid ant colony optimization algorithms. In: EvoCOP 2014 - 14th European Conference on Evolutionary Computation in Combinatorial Optimisation April 23-25, 2014. Granada, Spain (2014), in press in Lecture Notes in Computer Science
5. Cicirello, V.A., Smith, S.F.: Modeling GA performance for control parameter optimization. Morgan Kaufmann Publishers (2000)
6. Cicirello, V.A.: Non-wrapping order crossover: An order preserving crossover operator that respects absolute position. In: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation. pp. 1125–1132. ACM, New York, NY, USA (2006)
7. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6, 182–197 (2002)
8. Develin, M., Hartke, S.G.: Fire containment in grids of dimension three and higher. Discrete Appl. Math. 155(17), 2257–2268 (2007)
9. Falkenauer, E., Bouffouix, S.: A genetic algorithm for job shop. In: Proceedings of the 1991 IEEE International Conference on Robotics and Automation. pp. 824–829 (1991)
10. Fleischer, M.: The measure of pareto optima. applications to multi-objective metaheuristics. In: Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003. pp. 519–533. EMO'03, Springer-Verlag, Berlin, Heidelberg (2003)
11. Goldberg, D.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison Wesley (1989)
12. Goldberg, D.E., Lingle Jr., R.: Alleles, loci, and the traveling salesman problem. In: Grefenstette, J.J. (ed.) Proceedings of the First International Conference on Genetic Algorithms and Their Applications. pp. 154–159. Lawrence Erlbaum Associates, Publishers (1985)
13. Haghighi, A., Asl, A.Z.: Uncertainty analysis of water supply networks using the fuzzy set theory and NSGA-II. Engineering Applications of Artificial Intelligence 32, 270–282 (2014)
14. Hartnell, B.: Firefighter! an application of domination. In: 20th Conference on Numerical Mathematics and Computing (1995)
15. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. Springer, Berlin, Heidelberg (1994)
16. Mumford, C.L.: New order-based crossovers for the graph coloring problem. In: Proceedings of the 9th International Conference on Parallel Problem Solving from Nature. pp. 880–889. Springer-Verlag, Berlin, Heidelberg (2006)
17. Oliver, I.M., Smith, D.J., Holland, J.R.C.: A study of permutation crossover operators on the traveling salesman problem. In: Proceedings of the Second International Conference on Genetic Algorithms on Genetic Algorithms and Their Applications. pp. 224–230. Lawrence Erlbaum Associates Inc., Hillsdale, NJ, USA (1987)
18. Sadeghi, J., et al.: A hybrid vendor managed inventory and redundancy allocation optimization problem in supply chain management: An NSGA-II with tuned parameters. Computers & Operations Research 41, 53–64 (2014)
19. Syswerda, G.: Schedule optimization using genetic algorithms. In: Davis, L. (ed.) Handbook of Genetic Algorithms. Van Nostrand Reinhold, New York, NY (1991)
20. Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., da Fonseca, V.G.: Performance assessment of multiobjective optimizers: An analysis and review. IEEE Transactions on Evolutionary Computation 7, 117–132 (2002)