2021 43rd Annual International Conference of the
IEEE Engineering in Medicine & Biology Society (EMBC)
Oct 31 - Nov 4, 2021. Virtual Conference

Optimal Deployment in Emergency Medicine with Genetic Algorithm Exemplified by Lifeguard Assignments*

Jonas Chromik[1] and Bert Arnrich[1]

Abstract— **In emergency medicine, workforce planning needs to satisfy a number of constraints. There are hard constraints regarding qualifications and soft constraints regarding the wishes of the personnel. One instance of such a planning problem is the assignment of lifeguards at the coasts of the North Sea and the Baltic Sea in Germany. These lifeguards are volunteers and thus accounting for wishes is crucial while qualification constraints must be satisfied nevertheless. This paper presents a genetic algorithm that solves this problem with sub-second runtime. We compare this genetic algorithm to a brute force solution creating optimal solutions at the expense of larger runtime complexity. The genetic approach outperforms the brute force approach in terms of runtime when there are more than 3 places of deployment while consistently producing optimal solutions within less than 10 generations.**

I. Introduction

In Germany, the coasts of the North Sea and the Baltic Sea are guarded by voluntary lifeguards during the summer. These lifeguards are sent there by life-saving associations, among others the German Life-Saving Association ("Deutsche Lebens-Rettungs-Gesellschaft", DLRG). There are different *stations* where lifeguards can be deployed to. The DLRG is responsible for 83 stations [1]. Volunteers are deployed to said stations for at least a week. Within these stations are different *places* where lifeguards can fulfil their duty, such as on a boat or a lifeguard tower. The assignment of lifeguard personnel to concrete places can change from day to day and is subject to a variety of hard and soft constraints. Workforce planning is currently done manually by a team lead. This is a time-consuming and error-prone activity. Furthermore, manual assignment might cause resentment by the personnel since they might suspect unfair treatment. This is not just a problem with volunteer lifeguards but an overall problem in emergency medicine: Satisfying hard and soft requirements under conditions that change daily. This paper is concerned with these constraints and how assignments can be found to satisfy them taking volunteer lifeguarding as a tangible example.

The rest of this work is structured as follows. In section I we give an overview of the domain of lifeguarding in Germany as well as the personnel assignment problem. In section II we describe two approaches solving the problem. Furthermore, we describe the evaluation procedure used to test these approaches. In section III we present the evaluation results. Finally, in section IV, we discuss the results and summarise this work.

* This work was not supported by any organisation

[1] Hasso Plattner Institute, University of Potsdam, 14482 Potsdam, Germany firstname.lastname@hpi.de

A. Background

In this section, we give a brief overview of lifeguarding with special focus on the situation in Germany. Especially the qualification and certification system of lifeguards is relevant in the context of this work. All active lifeguards in Germany have a basic training and certification that is equivalent to the international ILS Lifesaver certificate, as defined by the International Life Saving Federation (ILSF) [2]. Furthermore, lifeguards can have additional qualification that are required for special tasks, i.e.: *Medics* provide extended medical assistance [3], *Rescue Boat Drivers* drive rescue boats [4] (equivalent to ILS Rescue Boat Driver [2]), and *Team Leads* ("Wachführer") lead the team of lifeguards and are responsible for a station [5]

These qualifications are relevant because certain tasks can only be done by sufficiently qualified personnel:

- Rescue boats need to be driven by a Rescue Boat Driver
- Lifeguard towers should be staffed with at least one medic to be able to provide sufficient medical assistance
- Team Leads should be stationed at the main tower since this tower provides the required management tools

According to these rules, an assignment needs to be made between personnel having a set of qualifications (example in Table I) and places requiring personnel with specific qualifications (example in Table II).

Person	Available Qualifications
Alice	Lifeguard, Medic
Bob	Lifeguard, Driver
Charlie	Lifeguard
Dave	Lifeguard, Team Lead
...	...

TABLE I: Example of lifeguard personnel with available qualifications.

Place	Required Qualifications
Tower 1 (main tower)	Team Lead, Lifeguard
Tower 2	Medic, Lifeguard
Tower 3	Medic, Lifeguard
Boat 1	Driver, Lifeguard
Boat 2	Driver, Lifeguard

TABLE II: Example of places for lifeguard deployment with required qualifications. For example, Boat 1 should be staffed with two people, one should be qualified as a driver and the other one should be qualified as a lifeguard.

Additionally, since the lifeguards are volunteers, wishes need to be taken into account. People might wish being to

be deployed together with a certain person (*person wishes*, example in Table III) or to be deployed to a specific place (*place wishes*, example in Table IV). Also, wishes are prioritized. While some wishes are extremely important to the involved people, other wishes might be only minor preferences.

Person A	Person B	Priority
Alice	Dave	0.3
Bob	Charlie	0.7

TABLE III: Example of person wishes: Persons A and B want to be deployed together with a priority between 0.0 and 1.0 (higher means more important).

Person	Place	Priority
Alice	Tower 2	0.5
Bob	Boat 1	1.0
Charlie	Tower 1	0.2

TABLE IV: Example of place wishes: A persons wants to be deployed to a specific place with a priority between 0.0 and 1.0 (higher means more important).

The artefact that needs to be generated from these lists of qualifications and wishes is an allocation of people to places (example in Table V) that satisfies all required qualifications with personnel having these qualifications and if possible accounts for wishes.

Place	Persons
Tower 1 (main tower)	Dave (as Team Lead), Charlie (as Lifeguard)
Tower 2	Alice (as Medic), Bob (as Lifeguard)
...	...

TABLE V: Exemplary allocation: Places get filled with sufficiently qualified personnel while accounting for wishes when possible.

One might raise the question of why this needs to be done algorithmically when the assignment problem could also be solved manually. First, this task is quite complex for a human as it involves integrating multiple sources of information and repeatedly checking for violations. Conversely, this is easily done algorithmically. Second, the problem needs to be solved daily, hence multiple times, with slightly changing inputs. Having an algorithm for this saves time and effort. Third, and most importantly to the authors, humans are susceptible to benefit oneself. A human doing the assignment might, consciously or unconsciously, prioritise its own wishes higher than the wishes of colleagues. And even if they do not, they might be suspected of such doing which, in turn, causes resentment with subsequent impairment of morale. To address these issues, we propose an impartial algorithmic solution to the described problem.

B. Related Work

Workforce planning is an immanent challenge faced by organizations with a non-trivial structure. Hence, the problem of allocating workforce to engagements is not a novel one. However, existing works mainly focus on matching required and available qualifications as hard constraints while not accounting for softer requirements like wishes [6]. This might be a reasonable strategy for regular employments with monetary compensation. However, organizations relying on volunteers which is common in emergency medicine in Germany can not count on such a rather naive approach ignoring the wishes of their workforce. There are works on satisfying hard constraints while optimizing soft constraints in terms of workforce planning. In [7], an evolutionary algorithm outperforms a mixed integer programming solver in such a task, which is why we also rely on an evolutionary/genetic approach in this case.

II. Methods

We solved the lifeguard assignment problem described in the previous section with two different approaches. First, we implemented a brute force algorithm evaluating every possible allocation and penalising for violations of qualifications and wishes. This algorithm serves as a generator for ground truth solutions since it deterministically finds the optimal allocation. Second, we implemented a genetic algorithm generating possibly slightly suboptimal solutions in a much shorter time. Before we explain both approaches in detail, we want to describe how penalisation is done.

A. Penalisation

We face the problem of having two different types of constraints: Matching available and required *qualifications* and fulfilling *wishes*. Qualifications are more important since tasks being done by insufficiently qualified personnel will create potentially dangerous situations which we want to avoid. This could be expressed using a generate-and-test pattern: Possible allocations are generated and subsequently tested for qualifications violations. While this would be sufficient for a brute force approach, generate-and-test does not fit the genetic approach. We, therefore, decided to use a penalisation system with two orders of magnitude. Allocations get assigned a penalty that describes how bad the allocation is (higher penalty means worse). Violations of wishes get penalised according to their priority, e.g. if a wish of priority 0.5 is violated, the penalty is increased by 0.5. Qualification violations are penalised by a higher order of magnitude so that no amount of fulfilled wishes can justify a qualification violation. For example, we might penalise a missing driver by increasing the penalty by 100.

B. Brute Force Approach

The brute force approach assesses all possible allocations, hence producing an optimal solution. This is achieved as follows:

1) Generate all possible allocations
2) Compute penalty for each allocation
3) Select the allocation with the lowest penalty as the final solution

This is, however, slow since all permutations of the list of personnel need to be created. Therefore, this algorithm has

a runtime complexity of $O(n!)$ where n is the number of lifeguards. Obviously, this does not scale. We can nevertheless use this algorithm to benchmark the genetic approach introduced in the next section.

C. Genetic Approach

Genetic algorithms start with a first-generation set of *individuals* (in our case allocations). For each individual, the *fitness* (the opposite of penalty) is computed and the fittest individuals are selected to pass into the next generation. While passing, the individuals might be changed by crossover (combining two individuals) or mutation (slight changes in a single individual). [8]

In our genetic approach to the lifeguard assignment problem, the initial individual generation produces random allocations, as shown in Table V. These allocations can include qualification and wish violations since they are randomly generated. However, the generation process ensures, that the allocations are consistent, i.e. no place is overfull or underfull and persons are not assigned to multiple places.

The fitness is assessed as described in subsection II-A: Qualification violations increase the penalty by 100 and wish violations are penalised according to their priority. It is important to notice here that fitness/penalty needs to be minimised which is contrary to the general notion of fitness. Thus, the individuals with the lowest fitness/penalty get selected to pass to the next generation.

Our algorithm does not use crossover, as combining two allocations while simultaneously ensuring consistency is not easily done. Therefore, mutation of individuals is the only means of altering allocations. Mutations ensure consistency by taking an existing allocation, selecting two people from it by random and swapping these people. For example, Charlie on Tower 1 is swapped with Bob on Tower 2, the result is that Bob is now on Tower 1 and Charlie on Tower 2.

D. Evaluation

Evaluation of the described approach is complicated since there are no publicly available data on lifeguard qualifications and wishes. To evaluate our solution anyway, we observed the lifeguards in Binz for two weeks to derive statistical properties which we can use to generate artificial data.

	35th week 2020[1]	36th week 2020[2]
total # of lifeguards (n)	23 (100%)	10 (100%)
# of medics (n_{medic})	16 ($\sim$ 70%)	6 (60%)
# of drivers (n_{driver})	5 ($\sim$ 20%)	3 (30%)
# of team leads (n_{lead})	2 ($\sim$ 10%)	1 (10%)
# of places to be staffed (m)	10	5
personnel-place ratio ($n : m$)	$\sim 2 : 1$	$2 : 1$

TABLE VI: Total numbers and percentages of qualifications during the observed period.

Using these percentages, we generated artificial data on available personnel, places to be staffed, and wishes. Input parameter for the data generator is the number of places the simulated station has (m). From this, we can derive the number of personnel (n) via the ratio between # of personnel and # of places, which is 2:1 ($n = 2 \cdot m$). Every artificially generated person has a 100% probability of having the lifeguard qualification since this is a prerequisite for being part of the team. The probability of an additional medic qualification is 50% which is slightly lower than stated in Table VI but from interviewing the lifeguards we learned that there are unusually many medics among them. The probability of an additional driver qualification is 25%, the average between the 35th and the 36th week. The probability of an additional team lead qualification is 10% as it is the case in both observed weeks.

For artificially generating places, we used the same percentages in order to account for stations being staffed according to their qualification. This is shown in Table VII.

Type of Place	Required Qualifications	# of occurrences
Main Tower	Team Lead, Lifeguard	$m_{\text{main}} = \lceil m \cdot 10\% \rceil$
Rescue Boat	Driver, Lifeguard	$m_{\text{boat}} = \lceil m \cdot 25\% \rceil$
Normal Tower	Medic, Lifeguard	$m_{\text{tower}} = m - m_{\text{main}} - m_{\text{boats}}$

TABLE VII: Types of places and how often they occur.

Person wishes (who wants to be together with whom?) are generated by randomly selecting persons (without replacement), generating pairs of these randomly selected persons, and assigning a random priority between 0.1 and 1. The number of person wishes is random between 1 and m.

Place wishes (who wants to be where?) are generated by randomly selecting persons (without replacement) and randomly selecting places (with replacement). Subsequently, pairs of persons and places are formed and a random priority between 0.1 and 1 is assigned. The number of place wishes is random between 1 and m.

III. Results

We evaluated our genetic approach (subsection II-C) against the supposedly much slower but in terms of results optimal brute force approach (subsection II-B) using data generated as described in subsection II-D. In this section, we show the results of this evaluation in terms of result optimality and runtime.

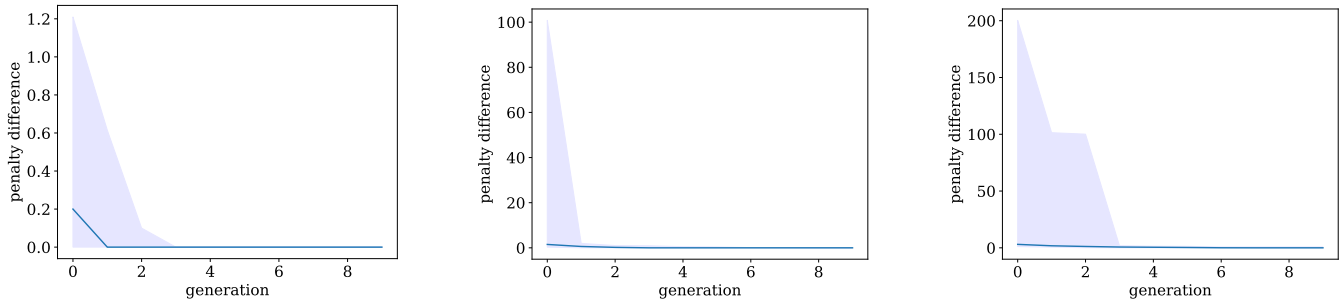
A. Optimality of Results

Figure 1 shows that the genetic approach generates optimal or close to optimal solutions for up to $m = 5$ in less than 10 generations with a population size of 100 individuals. This is achieved in less than 0.1 seconds on a Intel® Core™ i7-7700HQ CPU, as shown in Figure 2. Larger problem sizes are hard to evaluate since the runtime of the brute force approach increases in a factorial manner. Hence, problem sizes of $m > 5$ take very long time[3]. However, we can assume, that the genetic algorithm continues to find optimal or close to optimal solutions with few generations (< 100), since this is a continued finding in throughout $m \in \{3, 4, 5\}$.

[1]Last week in high season.

[2]First week in low season.

[3]Several minutes on the benchmark machine for $m = 5$.



(a) # of places $(m) > 3$ (b) # of places $(m) > 4$ (c) # of places $(m) > 5$

Fig. 1: These three plots show on the y axis the difference between the penalty on the optimal solution (as obtained via brute force) and the genetic solution after a number of generations, on the x axis. Population size is 100 individuals. Note the differently scaled y axes.

B. Runtime Comparison

We already noticed in subsection II-B that the brute force approach has a runtime complexity of $O(n!)$ or, respectively, $O(m!)$ since $n = 2 \cdot m$ (see subsection II-D) and constant factors are ignored in Big-O notation. Hence, the genetic algorithm is bound to outperform the brute force approach as problem size increases. Figure 2 shows that the break even point is between $m = 3$ and $m = 4$.

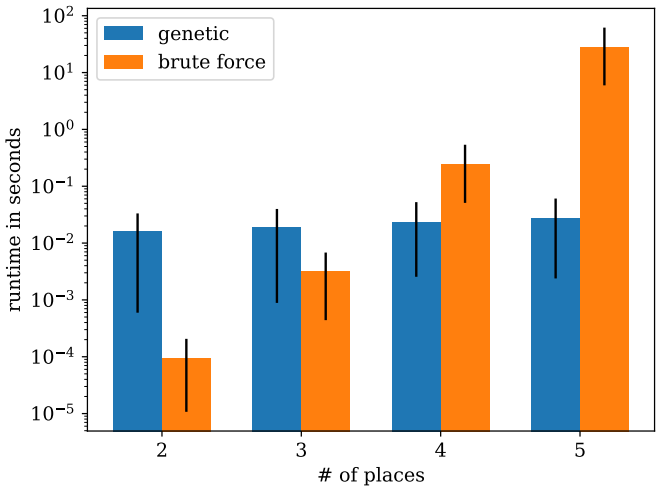


Fig. 2: Runtime comparison between the genetic and the brute force approach shows that the genetic approach is faster for assignment problems with # of places $(m) > 3$. The runtimes were obtained through genetically evolving 100 individuals over 10 generations. The y scale is logarithmic.

IV. Discussion

The proposed genetic algorithm solves the lifeguard assignment problem with optimal or close to optimal results in terms of qualifications and wishes. We showed this for assignment problems with up to $m = 5$ places of possible assignment and $n = 2 \cdot m = 10$ lifeguards and only refrained from larger problem sizes because the brute force algorithm solving as ground truth for optimal solutions would take too long. In terms of runtime, the genetic algorithm shows an approximately linear runtime complexity with respect to the problem size m. This is because the runtime of the genetic algorithm mainly depends upon the population size and the number of generations. Conversely, the runtime of the brute force algorithm increases factorially with the problem size. Consequently, the genetic algorithm is faster for $m \geq 4$.

A current limitation is that we do not account for additional qualification, such as advanced medical qualifications. Further work is needed to address this issue, for example by introducing additional qualifications as a soft constraint.

In summary, the genetic algorithm presented in this paper is sufficiently suitable to allocate lifeguards – or emergency medical personnel in general – to appropriate places accounting for required qualifications at the places and available qualifications of the lifeguards. In addition, wishes of the lifeguards in terms of where they want to be deployed and with whom they want to be deployed are also taken into account. The algorithm is sufficiently fast to be used in practice, even with large lifeguarding stations having many places and lifeguards. Furthermore, the usage of an algorithm for allocating lifeguards removes the risk of corruption, bribery, or favouring, as well as suspicions thereof.

References

[1] DLRG, "ZWRD-K – Wachstationsinfos." https://zwrd-k.dlrg.de/index.php?doc=stationInfosOffen.

[2] ILSF, "Equivalency Table Germany." https://www.ilsf.org/wp-content/uploads/2020/04/Equivalence-2020-24-Germany-DLRG.pdf.

[3] Deutsche Lebens-Rettungs-Gesellschaft e. V., "Ausbildungs- und Prüfungsordnung Medizin," 2017.

[4] Deutsche Lebens-Rettungs-Gesellschaft e. V., "Prüfungsordnung Bootswesen," 2018.

[5] Deutsche Lebens-Rettungs-Gesellschaft e. V., "Prüfungsordnung Wasserrettungsdienst," 2020.

[6] P. De Bruecker, J. Van den Bergh, J. Beliën, and E. Demeulemeester, "Workforce planning incorporating skills: State of the art," *European Journal of Operational Research*, vol. 243, pp. 1–16, May 2015.

[7] J. Peters, D. Stephan, I. Amon, H. Gawendowicz, J. Lischeid, L. Salabarria, J. Umland, F. Werner, M. S. Krejca, R. Rothenberger, T. Kotzing, and T. Friedrich, "Mixed Integer Programming versus Evolutionary Computation for Optimizing a Hard Real-World Staff Assignment Problem," *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 29, pp. 541–554, July 2019.

[8] D. Whitley, "A genetic algorithm tutorial," *Statistics and Computing*, vol. 4, pp. 65–85, June 1994.