

Genetic Algorithm for the Maximum Coverage Location Problem Applied to Medical Emergency

Matheus Ferraroni Sanches, Wellington Lobato Junior
UNICAMP

Cidade Universitária Zeferino Vaz - Barão Geraldo, Campinas - SP
[ferraroni, wellington]@lrc.ic.unicamp.br

Fábio Luiz Usberti, Leandro Aparecido Villas
UNICAMP

Cidade Universitária Zeferino Vaz - Barão Geraldo, Campinas - SP
[fusberty, leandro]@ic.unicamp.br

RESUMO

Com o crescimento das áreas urbanas, vários serviços precisam ser atualizados para manter uma boa qualidade. Uma área profundamente afetada são os Serviços de Emergência Médica (EMS), que se não for bom e rápido pode resultar em perdas de vidas. A localização onde os veículos de emergência estão tem uma influência significativa no tempo levado para que ele chegue até o local de uma emergência. Em ambientes complexos, como os de grandes cidades, a localização pode ter um impacto ainda maior no tempo gasto, já que congestionamentos e bloqueios podem rapidamente aumentar esse tempo. Este trabalho propõe uma abordagem que usa algoritmos genéticos para selecionar a localização das estações bases, onde veículos de emergência podem ficar, considerando informações de tráfego para determinar qual a área de alcance a partir de suas bases.

PALAVRAS CHAVE. Algoritmos Genéticos, Problema de Cobertura Máxima, Otimização.

Tópicos: PO na Área da Saúde

ABSTRACT

With the growth of urban areas, various services need to be updated in order to keep attendance with good quality. One profoundly affected area is the Emergency Medical Services (EMS), which if the service is not good and fast may result in loss of life. The location where the emergency vehicles are positioned has a significant influence on the time taken by the emergency vehicle to arrive at an emergency location. In a complex environment, as the large cities, the location may have an even more impact on that time, as congestion levels and road blocks may quickly increase the time taken. This work proposes an approach using a genetic algorithm to select the location of the standby sites where the emergency vehicles can stay, considering traffic information to determine the area that these vehicles can reach in a given time.

KEYWORDS. Genetic Algorithm, Maximum Coverage Problem, Optimization.

Paper topics: PO in the Health Area

1. Introduction

Emergency Medical Services organizations play an important role in modern health systems, and their ability to accurately respond to emergency calls can have a significant impact on patients health and recovery Andersen [2019]. EMS are critical elements of modern health systems, and they are responsible for the pre-hospital component of health systems. With the growth of urban centers and the number of vehicles present in them, the demands in various sectors are affected, such as food, safety, productivity, and mobility areas.

The technologies present in cities are being adopted more, these technologies range from a GPS sensor, capable of providing vehicle location in real time, to intelligent traffic management systems capable of improving traffic at peak times through intelligent traffic light control Kang et al. [2014].

Existing transport systems can take advantage of sensors available on vehicles and roads to gather information and make smarter decisions, such as selecting faster routes de Souza et al. [2018]. The set of subsystems that operate in the transport system is known as Intelligent Transport System (ITS) and offers advantages such as less polluting traffic, reduced congestion, and faster travel Cunha et al. [2017]. Services such as emergency management can take advantage of ITS to move their vehicles faster in cities.

Recent studies on the management of EMS show that there are three main areas: standby sites location, emergency vehicle reallocation, and emergency vehicle dispatch Bélanger et al. [2019]. The vehicle dispatch tries to find the best vehicle to dispatch to a specific demand. The reallocation problem is concerned about changing the vehicles between the standby sites to better attend the demands. The static location problem tries to find the best places to set standby sites in a given area.

The increasing urban growth raises essential needs to the cities, related to traffic congestion, safety, and mobility. The safety in the cities is a demand that is profoundly affected by diverse factors. The position of the standby sites where the emergency vehicles are parked has a significant impact on the time spent to arrive at an emergency location. If the transit on the route used by the emergency vehicle dispatched has a high level of congestion, this time can increase significantly.

This work proposes the result of using a Genetic Algorithm (GA) to deal with the static location problem, which is by itself an NP-hard problem [Farahani e Hekmatfar, 2009]. The data used to calculate the best location for the standby sites is their respective coverage area. Also, the areas that can be reached from the standby site are not circles or square areas, but a polygon that considers traffic information about a busy day at the rush hour. The polygonal coverage of each site increases the complexity of the calculations considerably, since if a standby site is moved a few meters from its original position the polygon may be substantially different.

The use of optimization models solves problems like the maximum coverage location are a good way to face them. During this work, we use a model inspired on evolution, the Genetic Algorithm. These kinds of algorithms store the important information in a data structure similar to a chromosome and apply different combinations and mutations in order to improve how good one element can solve the problem faced Whitley [1994].

The remainder of this paper is structured as follows. Section II outlines related works from the literature. Section III describes how the methodology was implemented and how it operates. Section IV discusses the scenario, evaluation and results obtained. Section V gives the conclusions of this paper.

2. Related work

Several works in the lasts years have been developed in order to improve how emergencies are handled, and efforts to reduce waiting time are possible through a better allocation of resources.

Toregas et al. [1971] was the first one to approach the emergency vehicle location considering the coverage. In this work, an area is considered covered if it can be reached by at least one vehicle within a time or distance. The main goal of this work is to minimize the number of vehicles that are used to cover the defined area. To perform its calculation, this work uses integer linear programming fed by a matrix of the minimum distance between any two locations.

In Nickel et al. [2016], the authors investigate the problem of choosing the location and number of ambulances and their bases in a specific region. The paper also shows that the possibility of considering just one sample of scenarios can lead to a misleading solution. The main goal of the work is to minimize the total cost for maintaining and installing these facilities but assuring a minimum coverage level. However, these works do not consider real information about the city, which may alter the results. Moreover, such approaches do not take into account the characteristics of the scenario.

Liu et al. [2016] introduce a new Double Standard Model (DSM) for EMS that works along with a GA to solve vehicle allocation problem that ensures acceptable service reliability with limited vehicle resources. The authors used DSM to deal with different types of vehicles and different priorities for the demands, using data from vehicles crashes in Chicago from 2004 to 2010. The proposed model requires that any demand site is covered by at least one EMS vehicle and the secondary standard with a portion of the covered sites achieving the service reliability requirement. This work aims to maximize the demand sites covered within the first standard meeting the required service reliability. However, Liu et al. only consider a limited quantity of standby sites, which means that the approach did not allocate enough resources to cover all the city. Besides, this approach did not evaluate the real behavior of the scenario.

Zhang e Li [2015] proposes a probabilistic model with chance constraints for locating and sizing emergency medical service stations. In the proposed model, the chance constraints are approximated as second-order cone constraints to overcome computational difficulties for practical applications. The proposed approximations associated with different estimation accuracy of the stochastic nature are meaningful on a practical uncertainty environment.

Based on the analysis of the related work, we conclude that it is essential to consider an approach using genetic algorithms to allocate the facilities in the best positions to reduce waiting time. This involves analyzing the behavior of the different data collected and optimizing the allocation based on the possible routes.

3. Genetic Algorithm Optimization

In this Section, we describe the proposed methodology, a greedy solution for this problem and the real world solution. In Section 3.1, we present the problem formulation for the maximum coverage location problem and how is the data acquisition from the city of Chicago. Section 3.2 describes the greedy approach employed to generate the initial population and also used to compare with the genetic-approach. Section 3.3 presents the genetic algorithm approach applied in the data acquired and how it selects the best individuals, in each generation.

3.1. Problem Formulation

The formulation assumes that a standby site can be placed anywhere in the city that has a road. This simplifies the process of checking if a location is available to set the standby site and allow the main goal of this work to be better explored. The number of emergency vehicles that should stay in each of the chosen locations is not calculated at this moment, only the area that can be covered is considered.

An area is considered covered if it can be reached by at least one standby site. If the same area can be reached by more than one standby site, this area is considered as overlap. Areas that

cannot be reached with the times used in this work were not removed, which means that lakes and parks are considered during the calculation of the coverage. Not removing these areas means that the coverage of 100% of the desired area is not possible, but this keeps the scenario closer to the real world. The number of standby sites is considered a decision variable in the formulation.

3.2. Greedy Approach

The greedy solution developed in this project checks all possible locations, and it attempts to fit a standby site. If there is a location with no overlap, it is selected. The solution found by this approach was enough to achieve coverage between 40% and 60%. Another approach, which randomly add standby sites and removed the ones with overlap, were also tested, but on the comparison between these greedy solutions, the first one provided a better solution and was chosen to compare with the proposed genetic algorithm.

3.3. Proposed Genetic Algorithm

As soon as the solution is initialized, there is a process of coordinate conversion which is done in order to simplify the mutation stage. As soon as this conversion is completed, the first pool can be created. Until this moment, the amount of standby sites needed to cover the city is unknown. To determine how many standby sites the first pool should be created, a simple greedy algorithm is used. This greedy algorithm keeps selecting locations to insert until there is no place left that can be selected without overlap. Once the element generated with the greedy algorithm is complete, a pool of random elements is generated using as a base the number of standby sites determined by the greedy algorithm. The greedy element is inserted into the pool with the new random elements.

At this point, the generation loop is initialized. If it is the first iteration, the pool will contain only the random elements and one element generated by one greedy algorithm. If it is one of the following generations, the pool will contain a percentage of new random elements and a percentage of elements generated based on previous generations.

The first step of the generation loop is to check if the stop criteria is satisfied. If this criteria is satisfied the execution is terminated. In this work, the stop criteria was set as 300 iterations of the generation loop.

The following step is to evaluate the fitness of each element. As the objective is to cover as much as possible with the least overlap as possible, the first step is to determine each standby site covered. This calculation can be viewed in line 24 of Algorithm 1, where the final area is divided by the initial area. The polygon containing the area of the city is restarted for every fitness calculation; this way the polygon can be updated and cropped, which is done every calculation. Once the coverage of a polygon is calculated, this area is subtracted from the area to cover, in this way areas that cover the same place are not counted.

The overlap between the standby sites in the same element is also computed and can be view in line 19 of Algorithm 1. The polygons in this calculation are not altered, as they will always be the same for the same standby site in the time used.

Once the coverage by the standby sites and their overlap are calculated, the fitness can be assigned to the solution. As shown in line 25 of Algorithm 1, if the overlap between the polygons is higher than the coverage, the fitness of that solution may be negative. As the first generation guarantees at least one solution with positive fitness, generated by the greedy algorithm, there will be at least one solution with positive fitness and no overlap. The parent selection guarantees this property in the new generation step.

For the parents' selection and the generation of a new pool of elements, the elements in the actual pool are ordered by higher fitness to low, which can be seen in line 6 of the Algorithm 2. Half of the elements with lower fitness score are eliminated from the next steps, in line 7, in order

Algorithm 1: Optimization Of The Maximum Maximum Coverage Problem

```

1 Input
2 areaToCover  $\leftarrow$  ChicagoBoundaries
3 K  $\leftarrow$  Time to consider that can be reached from a standby site
4 Output
5 bestSolutionFound is the best solution found during the generations.
6 begin
7   timeReach  $\leftarrow$  K
8   areaToCover  $\leftarrow$  coordinateConversion(ChicagoBoundaries)
9   greedyElement  $\leftarrow$  generateGreedySolution(timeReach)
10  amtStandbySites  $\leftarrow$  greedyElement.AmountStandbySites
11  pool  $\leftarrow$  generateFirstPool(amtStandbySites)
12  pool.add(greedyElement)
13  penalty  $\leftarrow$  PenaltyRate
14  while not (stopCriteria.satisfied) do
15    for element in pool do
16      areaToCover  $\leftarrow$  areaOfChicago
17      polygons  $\leftarrow$  getPolygonsFromElement(element)
18      n  $\leftarrow$  polygons.size
19      overlap  $\leftarrow$  ( $\sum_{i=1}^n \sum_{j=i+1}^n \text{overlap}(\text{polygons}_i, \text{polygons}_j)$ )
20      overlap  $\leftarrow$   $\frac{\text{overlap} * \text{penalty}}{n}$ 
21      while polygons  $\neq \emptyset$  do
22        areaToCover.updateArea(polygons[0])
23        polygons.pop(0)
24      coverage  $\leftarrow$   $\frac{\text{areaToCover.finalArea}}{\text{areaToCover.initialArea}}$ 
25      element.score  $\leftarrow$  coverage-overlap
26    pool  $\leftarrow$  createNewGeneration(pool)
27  bestSolutionFound  $\leftarrow$  getBestElement(pool)

```

to increase the mean fitness score and try to let the best elements to be used as the base for the next generations.

The rest of the elements are used to generate the next pool. Using the randomness rate, defined at the initialization, is possible to determine how many elements are generated with a completely random set of standby sites and how many must inherit from the actual generation. During the development of this work, the randomness rate was defined as 15%, which means that 15% of the elements in each generation, after the first one, will be randomly generated. Having a percentage of new random elements in each generation is a good way to insert variability in the current pool.

The remaining 85% of the elements that should be part of the next pool will inherit from an element of the current pool. To select which element will be used as the base to new ones, a probability vector was defined, which guarantee that the greater the fitness of an element, the higher the chance that this element will be used to generate new elements. In line 9 of Algorithm 2 a vector of probability is created to select these elements.

Once a new element is generated inheriting the genome from another element, the entire genome is copied and will be mutated later. There is a chance that the number of standby sites changes during this process. It can be randomly increased by one or decreased by one, as it can be viewed in line 14 of Algorithm 2. If the number of standby sites is increased by one, a random standby site is added at the end of the genome. If the number of standby sites is decreased, a standby site is removed from the genome, which may be a random base station or the standby sites with most overlap with other standby sites at that element.

When the number of standby sites changes from the parent, a secondary element is created, and this secondary element will not endure mutation. This is done in order to evaluate the parent element with change restricted to the number of base stations. This process can be seen in lines 18 to 20 of Algorithm 2.

The best elements from a generation are copied to the new generation, in order to guarantee that the best solutions found so far keep being considered during the next generations. This process can be seen in lines 23 and 24 of Algorithm 2.

Algorithm 2: Creation Of New Generation

```

1 begin
2   pool ← pool of elements from actual generation
3   newPool = []
4   extend ← 1 - randomness rate
5   limitElements ← Total elements per generation
6   pool.orderByScore()
7   pool.discardWorseHalf()
8   n ← pool.size
9   distribution ←  $[n, n - 1, n - 2, \dots, 1] \cdot \frac{1}{\sum_{i=1}^n i}$ 
10  i ← 0
11  while  $i < \text{limitElements} * \text{extend}$  do
12    i ← i+1
13    parent ← pickRandom(pool, distribution)
14    newStandbySites ← parent.standbySites + randomInt(-1,1)
15    element1 ← new Element(parent,newStandbySites)
16    element1.mutationActive()
17    newPool.add(element1)
18    if not (parent.standbySites==newStandbySites) then
19      element2 ← new Element(parent,newStandbySites)
20      newPool.add(element2)
21    while newPool.size() < limitElements*0.1 do
22      newPool.add(new Element(None, pool[0]))
23    i ← 0
24    while limitElements*0.1 do
25      newPool.add(pool[i]) i ← i+1
26  pool ← newPool

```

4. Computational Experiments

This Section presents the evaluation of the proposed genetic algorithm based solution for allocating standby sites. We evaluated the impact of different standby sites locations for better coverage while trying to reduce the overlap between the covered area by more than one standby site. We analyzed how the proposed approach works differently concerning the time required to reach the standby site. Moreover, we provide the optimal number and positions of standby sites for the city of Chicago.

4.1. Scenario Description

We used the API from HERE¹ to acquire the information used to calculate how far a vehicle can get from a standby site in a given time, considering the data of Chicago as a base scenario. The city of Chicago has been chosen since different data are available on this. All the data used is provided by the city hall. This allows us to reproduce the real behavior of the city dynamics according to empirical data. We conducted 300 iterations of the generation loop, and the randomness rate used during the tests was fixed at 15%. We set different parameters related with mutation rate during generations, time to reach from a standby site and seed. The programming language adopted was Python 2.7.

The data used in the experiments consider one day, reflecting the rush hour usually presented in big cities such as Chicago. We used four different times to compute the area that can be reached from a standby site. All the main simulation parameters are summarized in Table 1.

Table 1: Genetic Algorithm Parameters

Parameters	Value
Initial population	60
Selection	Rank-based selection
Encoding	Binary
Penalty rate	1000
Amount of elements	60
Mutation Rate	From 0.25 to 0.0075
Randomness rate	15%
Iteration limit	300
Time to reach an event (minutes)	4,5,6 and 7
Starting standby sites	Defined by greedy solution
Amount of elements discarded	50%
Amount of best elements reinserted	10%

4.2. Results

The allocation of standby sites, after the application of the genetic algorithm, can be visualized in Figure 1. In each image the result obtained after the interactions carried out by the methodology, changing only the minutes of reach from a standby site. It is possible to notice the relation between time and quantities of standby sites. As the number of standby sites decreases, the higher is the time reaching. Figure 1(a) is the one with most standby sites, as this one has the smaller time used. Figures 1(b) and 1(c) show that the amount of standby sites decreased quickly. Figure 1(d) is the one with the fewer standby sites, as it has the largest time between the tests performed.

According to the evolution presented in Figure 2 it is possible to see the changes during the generation phase. The evolution of the coverage during the generations is presented in Figure 2(a). This value is the mean coverage of the best creature of every seed used in the same generation.

¹www.here.com

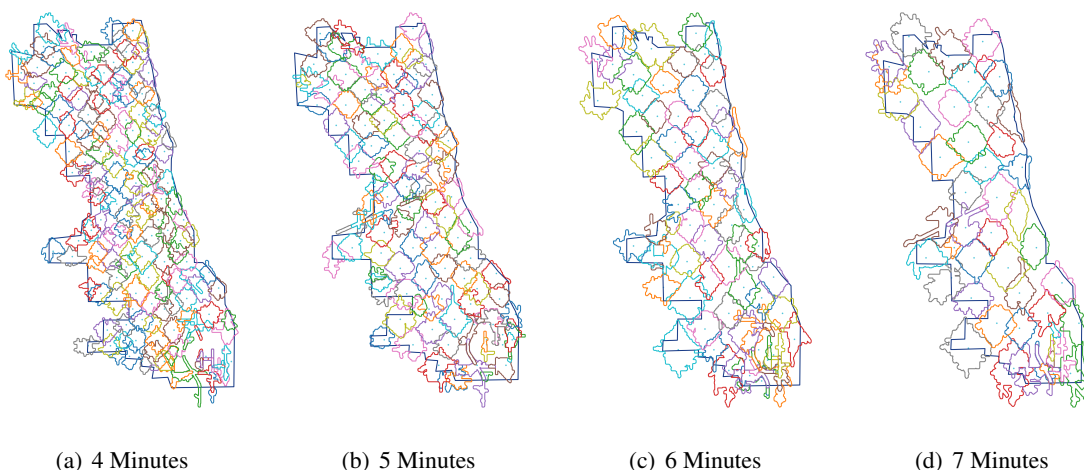


Figure 1: Different time to reach from the standby site

The starting coverage of all the scenarios was close to 35%, and the final values are close to 8%. A coverage close to 100% is a hard thing to accomplish with the times used, as the city has parks and lakes that can not be covered with the times used. Furthermore, when the time used in the scenario is more significant, the evolution tends to occur faster in the beginning, this can be explained by the addition of new standby site having a higher influence in the coverage calculation.

The changes in the amount of standby site are represented in Figure 2(b). Unlike the coverage, when the time used in the scenario is greater, the number of standby site tend to be reduced at the beginning. As this is defined by a greedy algorithm, with a smaller time being used, the polygons generated are smaller allowing more candidate polygons to be chosen for the same area. This results in a higher number of standby sites at the beginning for almost the same coverage. This difference is also reinforced at the end of the generations when the scenario with 4 minutes needs more than three times the standby sites than the scenario with 7 minutes to achieve almost the same Chicago coverage.

The overlap between the standby sites is one important metric, as it is used to calculate the fitness for each element in the pool. This information is represented in Figure 2(c). The initial overlap has a small variation, smaller than 7%. During the generations, the overlap changes quickly, but during all the generations it is smaller than 10% and tends to be minimized at the last generations. The minimization at the last generations is due to the optimization performed by the proposed methodology. Once the number of standby sites used is enough, small changes may help to reduce the overlap and increase fitness. The best fitness along with the generations never reduces, as the best element from the prior generation is copied to the actual generation. This behavior can be seen in Figure 2(d).

In Figure 3 the coverage of Chicago is compared between the chosen approaches, varying just the time to reach from the standby site. The coverage achieved by the proposed solution can keep close to 80% in almost all times. The real world has lower coverage when the time used is 4 minutes, but shows good performance when the times are 6 or 7 minutes. The greedy solution presented a lower fitness, with a significant difference, in all scenarios. To achieve 100% of coverage is a hard, if not impossible, task, as there are areas that are not covered within the times considered in the experiments.

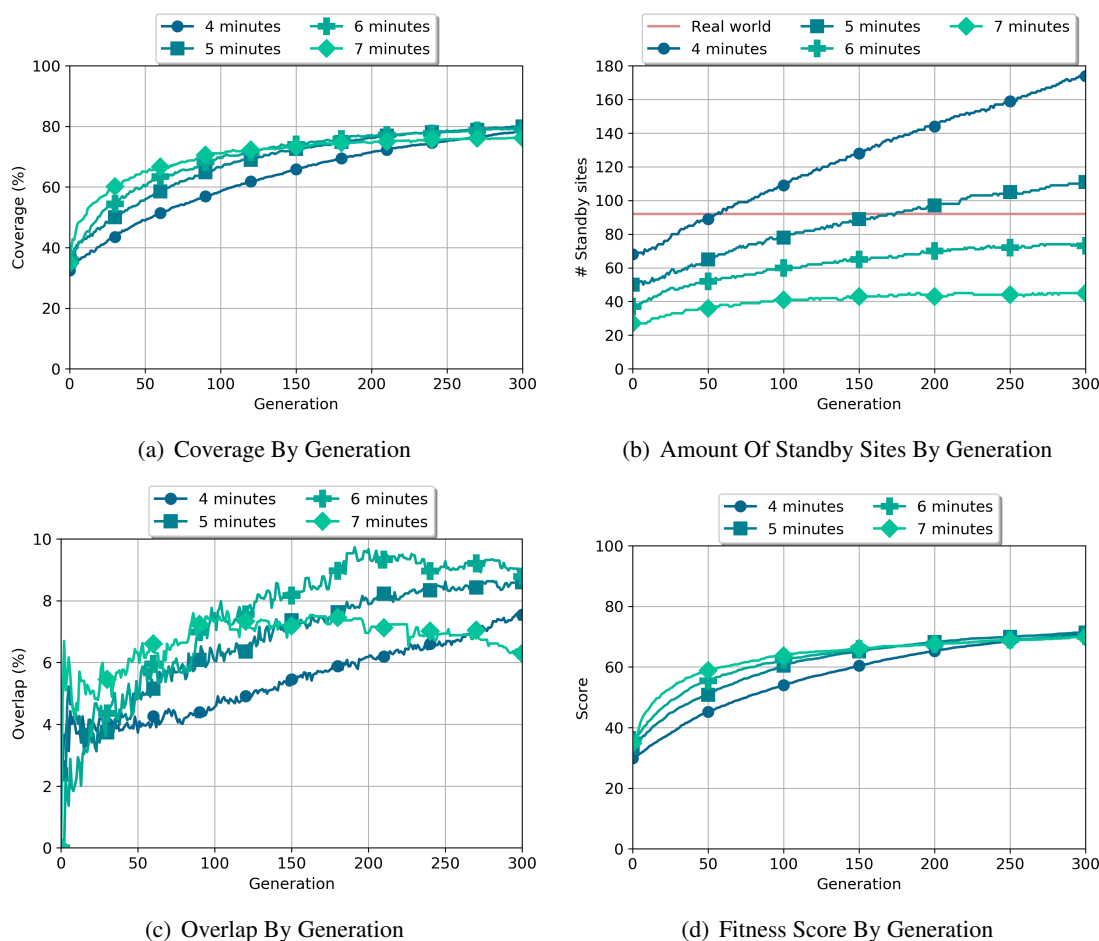


Figure 2: Evolution Over Generations

Figure 4 presented the overlap between the standby sites. The greedy solution was set to have no overlap during its calculation, so its data is not plotted. The mean average of the proposed solution is smaller than 10% for all scenarios, which is close to the real world only in the 4 minutes calculation. When the time increases, the real world has a quick increase in the overlap, achieving total overlap in the 7 minutes calculation. Even though the real world may show a slight improvement in the coverage with 6 and 7 minutes, the overlap increases way more than the proposed solution.

Alongside with coverage and overlap, it is possible to evaluate the number of standby sites used by each solution to achieve the presented results. Figure 5 shows the amount of standby sites used by the solutions. The data presented about the real world is fixed, as they are already in position. The data about the greedy solution is smaller in almost all scenarios evaluated, as this solution was set to have no overlap. The proposed solution starts with a higher number, but decrease quickly. Even though the number of standby sites was significantly smaller than the real world, this solution was able to keep a high coverage and slow overlap.

From our performance evaluation analysis, we identified that our genetic algorithm approach delivered standby sites with 17% more coverage of the city than the real disposal of facilities, and also reduced the overlap between the standby sites up to 95%. When the proposed approach is

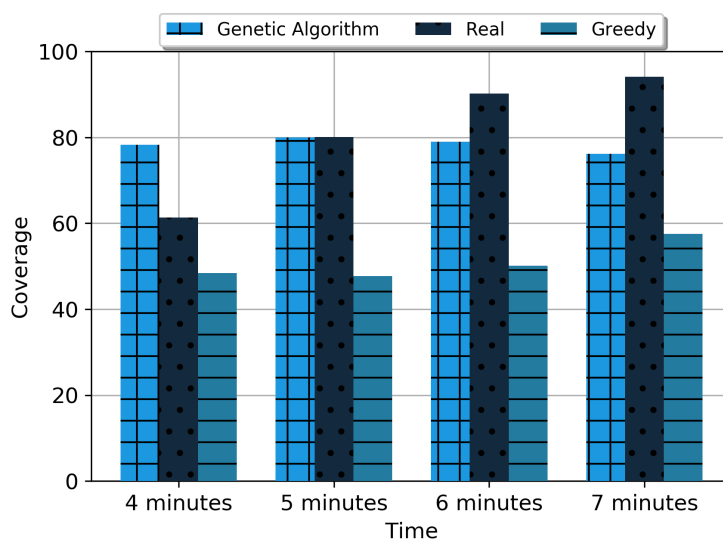


Figure 3: Coverage Comparison

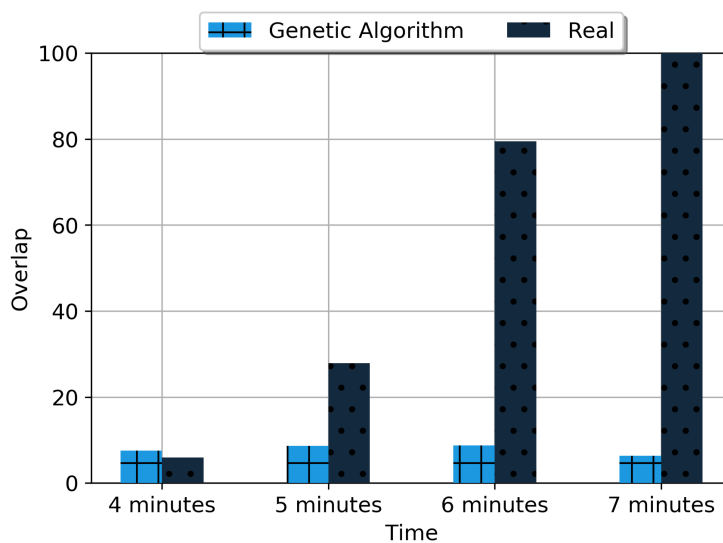


Figure 4: Overlap Comparison

compared with other solutions, the coverage difference increases even more, which occurs because the greedy solution is not capable of optimizing the overlap.

5. Conclusion and Future Work

This work introduced an approach using a genetic algorithm that considers traffic information to select the best position to allocate standby sites for ambulances in the city of Chicago. The genetic algorithm was able to consider the coverage polygons in order to fit them into the city boundaries while reducing the overlap between them. From our performance evaluation analysis, we identified that the approach solution was able to increase the cover of the city up to 17% more than the real disposal of standby sites, and also reduce the overlap between the standby sites up to 95%. This is because the genetic algorithm took into account the different characteristics of the collected data to allocate the standby sites in the most suitable places. The results show the efficiency

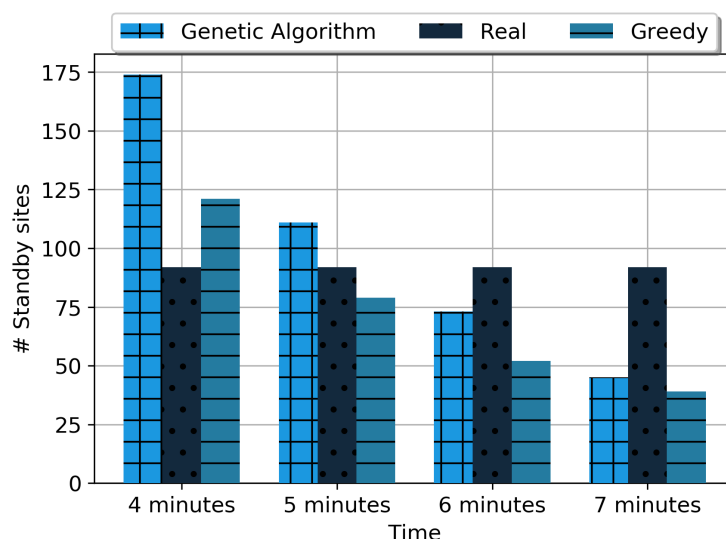


Figure 5: Amount of Standby Sites Comparison

of the approach compared to the real world concerning ensuring low servicing times for the health units.

As future works, we aim to consider a higher amount of data to compute the area that can be reached from a standby site within a given time. Also, a context-aware approach should be investigated, considering regions with different levels of crashes or even more populated regions. We also intend to evaluate the evolution of this approach in different scenarios, like other cities.

References

- Andersen, B. M. (2019). Ambulances, emergency medical service (ems) and other transports of patients. In *Prevention and Control of Infections in Hospitals*, p. 973–983. Springer.
- Bélanger, V., Ruiz, A., e Soriano, P. (2019). Recent optimization models and trends in location, relocation, and dispatching of emergency medical vehicles. *European Journal of Operational Research*, 272(1):1–23.
- Cunha, F. D., Maia, G., Celes, C. S. F. S., Santos, B. P., Rettore, P. H. L., Campolina, A. B., Guidoni, D., Souza, F. S. H., Ramos, H., Villas, L., Mini, R. A. F., e Loureiro, A. A. F. (2017). Sistemas de transporte inteligentes: Conceitos, aplicações e desafio. Mini curso.
- de Souza, A. M., Pedrosa, L. L. C., Botega, L. C., e Villas, L. (2018). itssafe: an intelligent transportation system for improving safety and traffic efficienc. *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*. ISSN 2577-2465. URL <https://doi.org/10.1109/VTCSpring.2018.8417760>.
- Farahani, R. Z. e Hekmatfar, M. (2009). *Facility location: concepts, models, algorithms and case studies*. Springer.
- Kang, W., Xiong, G., Lv, Y., Dong, X., Zhu, F., e Kong, Q. (2014). Traffic signal coordination for emergency vehicles. In *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, p. 157–161.

- Liu, Y., Li, Z., Liu, J., e Patel, H. (2016). A double standard model for allocating limited emergency medical service vehicle resources ensuring service reliability. *Transportation research part C: emerging technologies*, 69:120–133.
- Nickel, S., Reuter-Oppermann, M., e Saldanha-da Gama, F. (2016). Ambulance location under stochastic demand: A sampling approach. *Operations Research for Health Care*, 8:24–32.
- Toregas, C., Swain, R., ReVelle, C., e Bergman, L. (1971). The location of emergency service facilities. *Operations research*, 19(6):1363–1373.
- Whitley, D. (1994). A genetic algorithm tutorial. *Statistics and Computing*, 4(2):65–85. ISSN 1573-1375. URL <https://doi.org/10.1007/BF00175354>.
- Zhang, Z.-H. e Li, K. (2015). A novel probabilistic formulation for locating and sizing emergency medical service stations. *Annals of Operations Research*, 229(1):813–835.