Coalition Formation of Members based on Geographical Location by Genetic Algorithm

Anon Sukstrienwong

School of Inforation Technology and Innovation, Bangkok University, Pathumthani, Thailand

Abstract **– In the era of globalization, the rapid development of mobile technologies has shown the emergence of new forms of group formation. Groups formed of people connected by certain social relationships can be easily set up by online social networks. Additionally, the geographical locations of online users have become an important feature for the group formation. Accordingly, we proposed an algorithm to search of an optimal group formation based on users' location of latitude and longitude coordinates from the map using a heuristic search algorithm named Genetic algorithm (GA). The main object of the proposed algorithm is to arrange people with dissimilar positions into proper groups to have a mean distance between members within a group as short as possible. Furthermore, an empirical performance comparison of previous greedy and GA-based algorithms by simulation results is presented to verify the efficiency of the proposed algorithm.**

Keywords – Genetic algorithm, geographical locations, group formation, Location-based formation, optimization.

1. Introduction

Over the past several years, many researches in various fields have been conducted approaches for forming optimized groups in order to obtain the maximum benefit of the coalition.

DOI: 10.18421/TEM93-06
https://doi.org/10.18421/TEM93-06

Corresponding author: Anon Sukstrienwong,
School of Inforation Technology and Innovation, Bangkok University, Pathumthani, Thailand.
Email: anon.su@bu.ac.th

Received: 18 March 2020.
Revised: 08 July 2020.
Accepted: 16 July 2020.
Published: 28 August 2020.

Firstly, an online algorithm was developed to assemble potential team members to deal with tasks in which members possess different skills within a social network [1]. The algorithm is used to form teams of members in the online cases that are mostly satisfied with the required skills and provably approximate the optimal solution. Moreover, the fast growth of users of high-end mobile technologies has allowed new forms of group formation; therefore, it has increased a lot of interest in the research area of innovative mobile software. Noulas et al. [2] addressed that inferring friendships in a social network through the exploitation of location information such as user geographical co-occurrences has been suggested by Crandall et al. [3].

In addition, user locations in online social network has become important among Internet users, and its popularity has increased significantly with the launch of Foursquare in 2009 [4]. Foursquare is an independent location data platform for understanding how Foursquare users move through the real world, but a user's location updates can only be seen by authorized friends. However, previous works have found that geographical distance and closeness could influence online relationships and the online formation [5]. Several related works [1], [6], [7] addressed the formation problem in online team formation in social networks as well.

Significantly, in this work, we consider coalition formation of members on a different perspective. Since we aim to establish optimized groups of members while taking into consideration the geographical distances between members, the heuristic search algorithm is employed to search for the optimized solution that different members from a nearby geographical location could be grouped together. Hence, created sub-groups of members must be equal in terms of the average distance between members of those sub-groups. One essential problem of forming groups is how to form an optimal coalition. Especially the optimization problem in group formation has been proven to be NP-hard [8]. Hence, it is recommended to adopt a heuristic method to solve the optimization problem.

GA is an eminent adaptive strategy for solving several complex optimization problems. Therefore,

in this paper, we proposed the algorithm based on GA for establishing optimized location-based groups of members. The paper is divided into six sections including this introductory section. Following this section is the related works, we survey some works relevant to this paper which are optimization algorithms, location-based formation, and group formation by GA. Section 3 presents details of our proposed GA for group formation based on geographical location. Since designing the right scheme of encoding the genes is a crucial task for GAs, this section provides detail of how the problem is encoded and how the GA operations are applied. To verify the efficiency of the proposed algorithm, experimental results and performance comparison of previous greedy algorithms are demonstrated in Section 4. Section 5 illustrates the time complexity of the proposed algorithm. The last section details conclusion of this paper and proposes some future directions.

2. Related Works

2.1. Location-based Group Formation

Previous researchers have employed geological group formation in many real-world problems. For instance, Han et al. [8] demonstrated how the geographical location of developers is successfully employed to improve the performance of team formation in social coding sites. Michalak et al. [9] presented coalitions of customers based on geographical proximity in the e-marketplace. Also, Spry and Hedrick [10] presented the development of a velocity-based control methodology allowing vehicles to move as a group while maintaining a desired formation pattern of formations in terms of generalized coordinates. The central ideal of the work is to generate the formation equations of motion and control using generalized coordinates.

2.2. Optimization Algorithms

Theoretically, an optimization problem is determined by many researchers as the problem of searching for the best solution from all feasible solutions [11]. Moreover, a procedure which is used to search for an optimization problem is called an optimization algorithm. In designing of an optimization algorithm, it is necessary to address what the objective of the problem is. The main objective could be to minimize production costs or to maximize production efficiency. It will be performed repeatedly by comparing various solutions until an optimum or a satisfactory solution is found. In the real world, optimization has become a significant part of current computer-aided design activities. Most activities for searching an optimized solution among a number of candidate solutions have been carried out using stochastic optimization (SO) methods. Basically, optimization methods generate their path through the parameter space and use some "random factor" [12]. Among the SO algorithms, genetic algorithms (GAs) are a vigorous search technique that is most successfully used to search for the optimal solution of optimization problems. Therefore, many GA-based approaches have been introduced for solving several complex optimization problems. For example, Goldberg [13] is well known for employing GAs to solve complex optimization problems with high levels of success. Yokoyama et al. [14] proposed the model based on genetic algorithm for optimizing the flexible job-shop scheduling problem (FJSP). They concluded that the proposed approach outperforms the existing methods on particular problems with some conditions.

2.3. Group Formation by GA

Existing of the group formation in literature is found in computer sciences and related fields. Moreover, group formation problems have been considered as an NP-hard problem [6]. Han et al. [10] proposed a GA-based approach to construct teams of developers in social coding sites considering geographical location and communication costs. Alberola et al. [15] proposed a GA-based algorithm for group formation to organize older people into activity groups. The authors affirmed that the GA-based algorithm is able to search for near-optimal solutions to all proposed scenarios. De Lit et al. [16] used Grouping genetic algorithm (GGA), which is a special class of GAs, for regrouping small independent units of a factory in order to reduce transportation and maintenance cost. In addition, Cruz and Isotani [17] addressed that group formation is one of the most important approaches to help designing effective collaborative learning activities. The authors indicated that the inadequate formation of groups can make students less enthusiastic about group activities and learning, which hinders the learning process.

The most common methods for grouping members are to increase diversity within dissimilar groups or to minimize the difference between the generated groups [18]. Hence, the well-balanced groups can be formed by placing members in the suitable groups based on specific criteria, such as characteristics, skills, learning styles, and educational knowledge.

2.4. Geographical Distance

In mathematics, the Euclidean formula is used to find the distance between two points in the plane with coordinates (θ_1, ϕ_1) and (θ_2, ϕ_2), which is

calculated by the square root of the sum of the squared differences between the coordinates, see Eq. (1).

$$Distance((\theta_1, \phi_1), (\theta_2, \phi_2)) = \sqrt{(\theta_1 - \theta_2)^2 + (\phi_1 - \phi_2)^2} \quad (1)$$

Nevertheless, the Earth is round. This equation may not be reasonable for latitudes and longitudes because these coordinates are not in a Cartesian coordinate system. Hence, we need to calculate the distance of the curve along the surface of the Earth. For calculating the geographical distance of two points precisely in the Earth's surface, it is necessary to estimate the exact distance in a difference way. We need to converse the latitudes and longitudes into the Cartesian coordinates of the points. Therefore, we need to use a special formula named "Haversine formula" to calculate the exact distance between two points on the Earth given their longitudes and latitudes. Denoting the latitude by θ, the longitude by H ϕ and R is the earth's radius (6371 km). Let x_1 be located in (θ_1, ϕ_1) and x_2 in (θ_2, ϕ_2) The distance of two points using the Haversine formula can be presented as below, which is recommended by Veness [19]. Note that R is the earth's radius. Hence, we need to multiply 6371 km with $\pi/180$, $R = 6371 * \pi/180 \approx 111.2397$ km.

$$\Delta\theta = \theta_2 - \theta_1, \quad (2)$$
$$\Delta\phi = \phi_2 - \phi_1, \quad (3)$$
$$a = sin^2\left(\frac{\Delta\theta}{2}\right) + \cos(\theta_1) * \cos(\theta_2) * sin^2\left(\frac{\Delta\phi}{2}\right), \quad (4)$$
$$c = 2 * atan2(sqrt(a), sqrt(1-a)), \quad (5)$$
$$distance(x_1, x_2) = R * c. \quad (6)$$

3. The Grouping by GA based on Geographical Location

3.1. Optimization Algorithms

Let $M = \{m_1, m_2, \ldots m_n\}$ be a set of n members, who need to form groups based on their locations. We aim to exploit the geographical location factor to form the group of members; therefore, each member represents the standard geographical coordinate. Hence, a member m_i can be written as a specific vector of $m_i = (lat, lon)$, where lat is latitude and lon is longitude. If we want to arrange n persons into q groups, then the group size will be $p = \left\lfloor \frac{n}{q} \right\rfloor$. Let G be a nonempty set of formed groups denoted as $G = \{G_1, G_2, \ldots, G_q\}$, where $G_i \cap G_j = \varnothing$ for $\forall i \neq j$. That is, one member cannot be allocated to multiple groups. It should be noted that the number of members for established groups must be equal, as we need the equality of group formation. However, in some situations, equal group size cannot be made. For example, if 8 persons have to be divided into 3 groups, each group has at least $\left\lfloor \frac{8}{3} \right\rfloor = 2$ members. Hence, the third group should contain one additional member, which makes this group equal to $\left\lfloor \frac{8}{3} \right\rfloor + 1 = 3$ members.

3.2. Pseudocode of the Proposed Algorithm

Importantly, the main purpose of our GA algorithm is to allocate members into a proper group based on their location and minimize the geographical distance between any two members within the generated group. The strategy for the algorithm is to repeatedly employ the recombination and mutation genetic mechanisms on the population of candidate solutions, where the fitness function applied to all chromosomes encoded the possible solutions. Pseudocode of the proposed algorithm is presented as below. As can be seen, it starts by randomly initialized population.

Pseudocode:
Input: PopulationSize, ProbabilityMutation, MaxGeneration, TotalMember, GroupSize, NumberOfGroup
Output: BestSolution

Population←InitializePopulation(PopulationSize, ProblemSize)
FitnessFunction(Population)
Gen ← 0
Children ← ∅
While (Gen ≠ MaxGeneration)
  Children1 ← SwapMutation (Parent, ProbabilityMutation)
  Children2 ← ScrambleMutation(Parent, ProbabilityMutation)
  Children ← Children ∪ Children1
  Children ← Children ∪ Children2
  FitnessFunction(Children)
  Population ← Selection(Population, Children, PopulationSize)
  Gen←Gen+1
End
  BestSolution ← GetBestSolution(Populaiton)
Return (BestSolution)

3.3. Encoding of the Problem

An important part of this subsection is to describe the application of GAs, especially how the problem is encoded and how the GA operation is applied. In this work, each chromosome is represented as the string of integer values. As it is considered in establishing group of *n* members, the length of the chromosome is equal to the total number of members

as well. The value of the gene i^{th} represents the group to which a member m_i belongs to. Basically, it is a kind of permutation. The total number of permutations for generating q groups of n members is $\frac{n!}{(p!)^q * q!}$, where $p = \left\lfloor \frac{n}{q} \right\rfloor$. For example, if ten members are divided into two groups, an example of a chromosome encoding the group formation of ten members divided into two groups is presented in Fig. 1. As can be seen, allocating 10 members into two groups is a kind of permutation of "1111122222". Hence, there are $\frac{10!}{(5!)^2 * 2!} = 126$ ways of forming these groups. Hence, chromosome1 presents the group formation of $G_1 = \{m_2, m_3, m_6, m_8, m_9\}$ and $G_2 = \{m_1, m_4, m_5, m_7, m_{10}\}$, and chromosome2 encodes the group $G_1 = \{m_1, m_3, m_5, m_6, m_7\}$ and $G_2 = \{m_2, m_4, m_8, m_9, m_{10}\}$.

3.4. Fitness Function

As stated previously, the main aim of our algorithm is to form dissimilar members into groups regarding their geographical locations, where a distance between members within a group is as short as possible. Let all groups have p members. That is, $|G_i| = p, 1 \leq i \leq p$. Then, $G_i = \{g_i^1, g_i^2, \dots, g_i^p\}$, where $1 \leq j \leq p$ and $g_i^j \in M$. Let us represent group G_i as a graph, see Fig. 2. A vertex represents a location of member, and there is a dotted connection between every two members. Since we need to calculate a distance between any two members, namely g_i^r and g_i^s where $r \neq s$ this undirected graph becomes a complete graph on p vertices, K_p, and each vertex has degree $p-1$. The mean distance of all vertices in K_p associated with Eq. (6) can be expressed in Eq. (7). As we assume that there are q different groups to be formed, the summation of all groups' mean distance can be presented in Eq. (8). As a chromosome used in this paper encodes all formed groups of members, we use Eq. (8) as a fitness function of a chromosome. Then, a fitness function of a chromosome is expressed as Eq. (9). Since the algorithm aims to establish optimized group formation by minimizing the average distance among members of generated groups, the chromosome with lowest fitness value is considered the best one. Moreover, each individual is assigned a fitness value derived by this equation.

$$AvgDist(G_i) = \frac{\sum_{r=1}^{q-1} \sum_{s=r+1}^{q} distance(g_i^r, g_i^s)}{q}, \quad (7)$$

$$\sum_{i=1}^{p} AvgDist(G_i) = \sum_{i=1}^{p} \left(\frac{\sum_{r=1}^{q-1} \sum_{s=r+1}^{q} distance(g_i^r, g_i^s)}{q} \right), \quad (8)$$

$$Fitness(chromosome) = \sum_{i=1}^{p} AvgDist(G_i). \quad (9)$$

3.5. Mutation Operator

Due to the chromosomal structure designed to encode the group formation for our problem, we therefore avoid using crossover operators. Basically, crossing between two different chromosomes always creates offspring encoding groups that do not have the same number of members as we are expecting. As the results show, it causes the formation of unbalanced groups. Hence, crossover operation is not necessarily performed in our algorithm. For better understanding, Fig. 3(a) presents an example of how the crossover operator may cause such a problem. In a common crossover operator, two given parents swap part of their chromosome creating two offspring. As we can see, offspring1 encodes the group formation of $G_1 = \{m_1, m_3, m_5, m_6, m_8, m_9\}$ and $G_2 = \{m_2, m_4, m_7, m_{10}\}$ and offspring2 encodes the group formation of $G_1 = \{m_2, m_3, m_6, m_7\}$ and $G_2 = \{m_1, m_4, m_5, m_8, m_9, m_{10}\}$. We can see that both of these chromosomes consist of groups, where $|G_1| \neq |G_2|$.

	1	2	3	4	5	6	7	8	9	10
Chromosome1	2	1	1	2	2	1	2	1	1	2
Chromosome2	1	2	1	2	1	1	1	2	2	2

Figure 1. Examples of encoded chromosomes, creating a group of ten members.

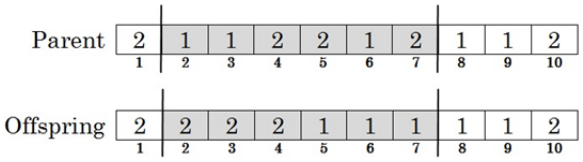
Figure 2. Example of a graph showing the connection of members of group G_i, where $g_i^j \in G_i$ and $1 \leq j \leq |G_i|$.

	1	2	3	4	5	6	7	8	9	10
Parent1	2	1	1	2	2	1	2	1	1	2
Parent2	1	2	1	2	1	1	1	2	2	2
Offspring1	1	2	1	2	1	1	2	1	1	2
Offspring2	2	1	1	2	2	1	1	2	2	2

(a) Example problem of using crossover operator.

	1	2	3	4	5	6	7	8	9	10
Parent	2	1	1	2	2	1	2	1	1	2
Offspring	2	2	1	2	2	1	1	1	1	2

(b) Swap mutation.



(c) Swap mutation.

Figure 3. Example of breeding operators.

Due to this limitation, mutation plays an important role to generate offspring for the next generation as it helps to maintain diversity in the genetic population. Hence, we employ two of the most commonly used mutations, which are swap mutation and scramble mutation in our algorithm. Both of them will proceed according to value of the mutated probability.

3.5.1. Swap Mutation

It starts by randomly selecting two positions on the chromosome and interchanging their values. An example of generating offspring by the swap mutation operation is presented in Fig. 3(b). Parent chromosome encodes the formation of $G_1 = \{m_2, m_3, m_6, m_8, m_9\}$ and $G_2 = \{m_1, m_4, m_5, m_7, m_{10}\}$. The generated chromosome is expressed in the formation of $G_1 = \{m_3, m_6, m_7, m_8, m_9\}$ and $G_2 = \{m_1, m_2, m_4, m_5, m_{10}\}$.

3.5.2. Scramble Mutation

It picks a subset of genes by selecting two positions on a chromosome at random. Then, their values between them are shuffled randomly making a new offspring. An example of using the scramble mutation operation is demonstrated in Fig. 3(c). The generated chromosome encoded the formation of $G_1 = \{m_5, m_6, m_7, m_8, m_9\}$ and $G_2 = \{m_1, m_2, m_3, m_4, m_{10}\}$.

3.6. Selection Operator

In this paper, a steady-state selection is used. It is a simple method of selecting chromosomes for the next successors. The main idea of this selection is that a big part of the current population will be selected for the next generation. After the reproduction phase using the mutation operators, the current population and descendants will be combined and then sorted in ascending order according to their fitness value. For simplicity, we use a selection sort to do this sorting. Subsequently, the best chromosomes, which are equal to the population size of chromosomes (M), are selected as the population of the next generation. Keep in mind that the algorithm aims to generate optimized group formation by minimizing the average distance among members of created groups. By doing this selection, the individuals with low fitness values, relative to the whole population, and high probability to survive are carried forward to the next generation.

4. Experimental Results and Discussion

In this section, we provide some experimental results to measure the efficiency of our GA-based approach. The proposed algorithm is implemented in C++, and the experiments were run in an Intel® Core™ i7 CPU 870 @ 2.93GHz machine, with 8GB of RAM.

Table 1. Parameters of genetic algorithms

Parameters	Meaning	Value
M	Population size of chromosomes	500
Gen_{Max}	Maximum number of generations	1000
p_m	Probability mutation	0.25

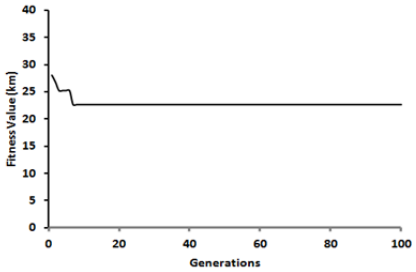


Figure 4. The best chromosome obtained by the proposed GA algorithm.

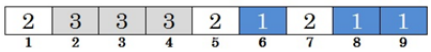
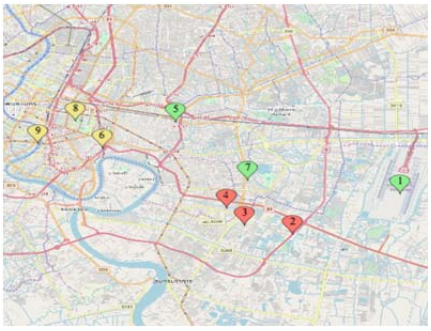


Figure 5. The best chromosome obtained by the proposed GA algorithm (9 members).



(a) Groups formed by the proposed algorithm

(b) Formed groups by the greedy algorithm

Figure 6. Experimental results of case study 1.

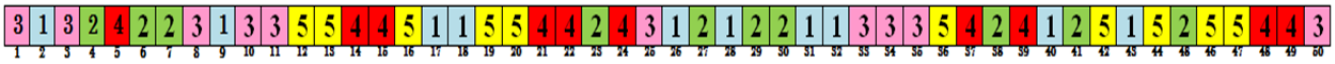


Figure 7. The chromosome obtained by the proposed GA algorithm (50 members).

In addition, we have used two empirical data sets for demonstrating the efficiency of the algorithm. The parameters certainly affect the performance of the GA; nevertheless, the value of these parameters is not the focus of this work. We used 500 (M) individuals in our population. And, probability mutation (pm) is 0.25. Also, we obtained good results using a population with 1000 individuals. Parameters of the GAs are given in Table 1.

4.1. Case Study1

Suppose there are nine members $(n = 9)$, scattered in different locations. Let us separate members into 3 groups, each group consisting of $\left\lfloor \frac{9}{3} \right\rfloor = 3$ $(q = 3)$ members. Hence, there are $\frac{9!}{(3!)^3 * 3!} = 280$ different ways of forming groups. Based on the simulation results, the proposed algorithm can generate the best chromosome within a few generations (Gen = 7) presented in Fig. 4., which can encode the group formation in the chromosome as demonstrated in Fig. 5. As can be seen on the map presented in Fig. 6(a), all members of each established group are located nearby, where $G_1 = \{m_6,m_8,m_9\}$, $G_2 = \{m_1,m_5,m_7\}$, and $G_1 = \{m_2,m_3,m_4\}$. Additionally, the fitness value of this chromosome is approximately 22.68 km. We also employ a greedy algorithm to form the group of nine members. The greedy algorithm can search for the same result; however it will yield arbitrarily bad results as well. This is because the greedy algorithm randomly picks the first member of the group and then immediately selects the best members who are located nearby. One of the results received by the greedy algorithm is illustrated in Fig. 6(b), where $G_1 = \{m_3,m_4,m_7\}$. $G_2 = \{m_1,m_2,m_9\}$, and $G_1 = \{m_5,m_6,m_8\}$. Besides, the average result obtained by the greedy algorithm is about 25.72 km, which is absolutely higher than the fitness value of the proposed algorithm.

4.2. Case Study2

To show that the proposed algorithm can work well with a large number of members, the proposed algorithm is tested on a dataset with forty (n=50) members, where their current location of latitude and longitude are randomly generated. Let's divide all members into 5 groups. Hence, each group consists of $\left\lfloor \frac{50}{5} \right\rfloor = 10$ $(q = 10)$ members. Hence, the search space is large because there are $\frac{50!}{(5!)^{10} * 10!} \approx 1.35E + 37$ different ways of forming these groups. Based on the simulation results, the best chromosome generated by the algorithm is demonstrated in Fig. 7., which can be decoded by group formation as below.

$G_1=\{m_2, m_9, m_{17}, m_{18}, m_{26}, m_{28}, m_{31}, m_{32}, m_{40}, m_{43}\}$,
$G_2=\{m_4, m_6, m_7, m_{25}, m_{27}, m_{29}, m_{30}, m_{38}, m_{41}, m_{48}\}$,
$G_3=\{m_1, m_3, m_8, m_{10}, m_{11}, m_{25}, m_{33}, m_{34}, m_{35}, m_{50}\}$,
$G_4=\{m_5, m_{14}, m_{15}, m_{21}, m_{22}, m_{24}, m_{37}, m_{39}, m_{48}, m_{49}\}$,
$G_5=\{m_{12}, m_{13}, m_{16}, m_{19}, m_{20}, m_{36}, m_{42}, m_{44}, m_{46}, m_{47}\}$

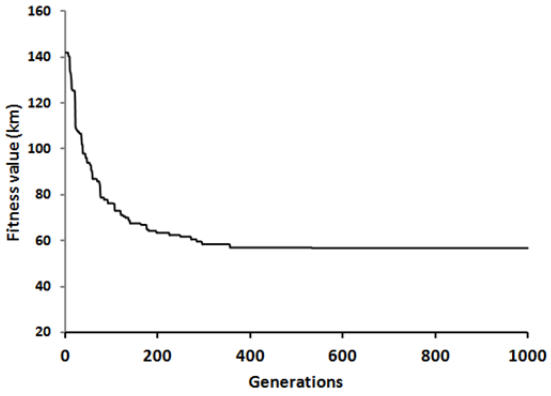
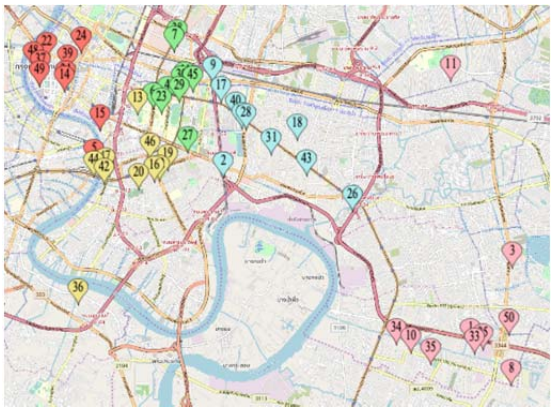
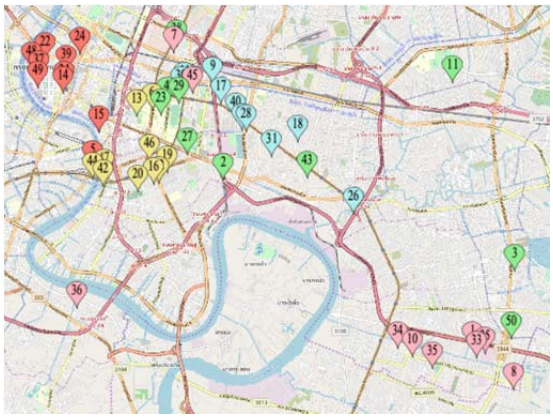


Figure 8. The experimental result obtained by proposed GA algorithm



(a) A result obtained by GA



(b) A result obtained by the greedy algorithm.

Figure 9. Experimental results of case study 2

Fig. 8. is the graphical result presenting the fitness value for each generation obtained by the GA algorithm. Obviously, the best fitness value is about 57.36 km, when the number of generations is approximate 400. Fig. 9(a) shows how each group is formed with members who are located in the same position on the map, making the average distance between members within formed groups as short as possible. Besides, we employed the same dataset with the greedy algorithm. The average result obtained by the greedy algorithm is about 92.10 km, which is absolute higher than the GA's result. However, if we allow the greedy algorithm to run several times, the best result obtained by the greedy algorithm is 57.36 km as well. An example result received by the greedy algorithm is demonstrated in Fig. 9 (b). It can be seen that some groups are quite similar to the group formed by the GA. Nevertheless, some groups consist of members who are far away.

5. Time Complexity of the Algorithm

Based on the pseudocode of our GA-based algorithm presented in the previous section, the time complexity depends on several factors. As can be seen, the algorithm has working processes that need to repeat until the predefined value of Gen_{Max} is met. Each process deals with a predefined number of a population of chromosomes, which is equal to M. Regarding the chromosome structure, the length of a chromosome equals the number of members, which is equal to n. Hence, the time complexity of the algorithm is $O(Max_{Gen}(nM))$. Moreover, the fitness evaluation and sorting are required to do for all chromosomes for each generation. The time complexity of the fitness function, which is presented in Eq. (7) – Eq. (9), is $O(pq^2)$. If all groups are equally formed with the same number of members, then $q = \frac{n}{p}$. The time complexity of the fitness function becomes $O\left(p\left(\frac{n}{p}\right)^2\right) = O\left(\frac{n^2}{p}\right)$. If p is small in value, we simply say that the running time of the fitness function is $O(n^2)$. The sorting method used in the selection operator also affects the time complexity. Basically, the complexity of sorting algorithms is $O(n^2)$. Therefore, the complexity of the algorithm is on the order of $O(Gen_{Max}(nM + n^2 + n^2))$. However, we can ignore the both Gen_{Max} and M, as they can be considered as a constant. That is, they are insignificant when compared to n. Finally, the algorithm has $O(n^2)$ time complexity. The GA-based algorithms and the greedy algorithm achieve the goal in forming groups with the intent of finding a global optimum. However, the GA-based algorithm is better than the greedy algorithm, since the greedy algorithm runs in $O(n^3)$ time complexity.

6. Conclusions and Future Work

The main objective of this paper is to propose a GA-based approach to search for optimal group formation based on members' geographical location. We exploit the Haversine formula to calculate the accurate distance over the earth's surface using the latitude and longitude between any two members within the generated group. Based on empirical case studies, the results illustrate the effectiveness of our proposed approach, which has been proven to be an effective approach of searching for optimal results. Additionally, it has been mathematically proven that our GA-based approach is more efficient than the greedy algorithm, since the proposed approach and the greedy algorithm has time complexity O(n^2) and O(n^3) respectively.

In our future work, we plan to investigate the algorithm for group formation based on the geographical locations of online users on mobile devices in order to form groups dynamically. Furthermore, we will also employ the proposed approach for forming group's members in organizations to strengthen the teams' relationship, cooperative learning, and boost groups' performance toward the required outcome.

References

[1]. Anagnostopoulos, A., Becchetti, L., Castillo, C., Gionis, A., & Leonardi, S. (2012, April). Online team formation in social networks. In *Proceedings of the 21st international conference on World Wide Web* (pp. 839-848).

[2]. Noulas, A., Scellato, S., Mascolo, C., & Pontil, M. (2011, July). An empirical study of geographic user activity patterns in foursquare. In *Fifth international AAAI conference on weblogs and social media*.

[3]. Crandall, D. J., Backstrom, L., Cosley, D., Suri, S., Huttenlocher, D., & Kleinberg, J. (2010). Inferring social ties from geographic coincidences. *Proceedings of the National Academy of Sciences*, *107*(52), 22436-22441.

[4]. Li, N., & Chen, G. (2010). Sharing location in online social networks. *IEEE network*, *24*(5), 20-25.

[5]. Brown, C., Nicosia, V., Scellato, S., Noulas, A., & Mascolo, C. (2012, May). Where online friends meet: Social communities in location-based networks. In *Sixth International AAAI Conference on Weblogs and Social Media*.

[6]. Lappas, T., Liu, K., & Terzi, E. (2009, June). Finding a team of experts in social networks. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 467-476).

[7]. Gaston, M., Simmons, J., & DesJardins, M. (2004, July). Adapting network structure for efficient team formation. In *Proceedings of the AAAI 2004 fall symposium on artificial multi-agent learning*.

[8]. Han, Y., Wan, Y., Chen, L., Xu, G., & Wu, J. (2017, May). Exploiting geographical location for team formation in social coding sites. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining* (pp. 499-510). Springer, Cham.

[9]. Michalak, T., Tyrowicz, J., McBurney, P., & Wooldridge, M. (2009). Exogenous coalition formation in the e-marketplace based on geographical proximity. *Electronic commerce research and applications*, *8*(4), 203-223.

[10]. Spry, S., & Hedrick, J. K. (2004, December). Formation control using generalized coordinates. In *2004 43rd IEEE Conference on Decision and Control (CDC)(IEEE Cat. No. 04CH37601)* (Vol. 3, pp. 2441-2446). IEEE.

[11]. Alkaim, A. F., & Al_Janabi, S. (2019, April). Multi objectives optimization to gas flaring reduction from oil production. In *International conference on big data and networks technologies* (pp. 117-139). Springer, Cham.

[12]. Alotto, P. G., Eranda, C., Brandstatter, B., Furntratt, G., Magele, C., Molinari, G., ... & Richter, K. R. (1998). Stochastic algorithms in electromagnetic optimization. *IEEE Transactions on Magnetics*, *34*(5), 3674-3684.

[13]. Goldberg, D. E. (2006). *Genetic algorithms*. Pearson Education India.

[14]. Yokoyama, S., Iizuka, H., & Yamamoto, M. (2015). Priority Rule-Based Construction Procedure Combined with Genetic Algorithm for Flexible Job-Shop Scheduling Problem. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, *19*(6), 892-899.

[15]. Alberola, J. M., del Val, E., Costa, A., Novais, P., & Julian, V. (2018). A genetic algorithm for group formation in elderly communities. *AI Communications*, *31*(5), 409-425.

[16]. De Lit, P., Falkenauer, E., & Delchambre, A. (2000). Grouping genetic algorithms: an efficient method to solve the cell formation problem. *Mathematics and Computers in simulation*, *51*(3-4), 257-271.

[17]. Cruz, W. M., & Isotani, S. (2014, September). Group formation algorithms in collaborative learning contexts: A systematic mapping of the literature. In *CYTED-RITOS International Workshop on Groupware* (pp. 199-214). Springer, Cham.

[18]. Nand, R., Sharma, A., & Reddy, K. (2018, December). Skill-based group allocation of students for project-based learning courses using genetic algorithm: Weighted penalty model. In *2018 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE)* (pp. 394-400). IEEE.

[19]. Veness, C. (2010). Calculate distance, bearing and more between Latitude/Longitude points. *Movable Type Scripts*, 2002-2014.