WORKSHOP TOUR SCHEDULING PROBLEMS USING GENETIC ALGORITHM

Akash Singh[*], Darshana Bhogesara and Nita H Shah[1*]
[*]Department of Applied Mathematical Science, Gujarat University, India.
[**]Department of Mathematics, Gujarat University, India.

ABSTRACT
Scheduling tasks in projects is a complex subject and one of the primary challenges in project management due to its high complexity. Any solution, whether optimal or near-optimal, must be periodically evaluated as projects grow to respond to changing situations. In this article, a genetic algorithm (GA) is applied to the scheduling problem to find the best optimal or near-optimal solution for scheduling. In the workshop tour scheduling problem, we aim to optimize the day assigned to each family to attend the workshop, thereby minimizing additional expenses. Therefore, we use a genetic algorithm to find the optimal solution for the workshop tour schedule.
KEYWORDS: Genetic Algorithm (GA), workshop tour problem, scheduling, fitness function, optimization

MSC: 90B35, 65Yxx

RESUMEN
Programar tareas en proyectos es un tema complicado y es uno de los principales problemas de la gestión de proyectos debido a su gran complejidad. Cualquier solución, ya sea óptima o casi óptima, debe evaluarse periódicamente a medida que los proyectos crecen para reaccionar ante situaciones cambiantes. En este artículo, se aplica un algoritmo genético (GA) al problema de programación para encontrar la mejor solución óptima o casi óptima para la programación. Para el problema de programación de En la programación del recorrido por el taller de Robin, Robin quiere optimizar qué día se le asigna a cada familia para asistir al taller para minimizar cualquier gasto adicional. Entonces, utilizamos un algoritmo genético para encontrar la solución óptima para el programa del recorrido del taller.
PALABRAS CLAVE: Algoritmo genético (AG), programación, función de aptitud, problema de recorrido de taller, optimización

1. INTRODUCTION

The workshop tour scheduling problem is an NP-hard combinatorial optimization problem, which has wide applications in the real world. The tour workshop scheduling problem arises when a center or organization offers guided workshops or tours over a fixed number of operating days and must assign visiting groups to specific days in an efficient and fair manner. Each group has preferences regarding when they would like to attend, while the center faces operational constraints and costs that depend on daily attendance levels.

Consider the problem of workshop tour scheduling. We consider that there are a certain number of days D where a center is open and groups of clients that have ranked q $(q<D)$ most preferred days to visit it. The scheduling is to assign a day to each group, knowing that the number of visitors shall be in a certain interval each day. Since the preferences can be not be always attained, there is a penalty function that measures the discomfort of the group if the assigned day is not the desired one. For those cases the center will gives one gift card from a set of options. The center has also an estimate of the cost of the service per day depending on the amount of visitors. The goal is to assign a day to each family in such a way that the costs of the visit and the penalty for the unsatisfied preference is minimized.

This combinatorial problem can be solved using a genetic algorithm. It is a computational optimization technique based on nature-inspired evolution and it is used to find optimal or near-optimal solution of many combinatorial problems. In the case of optimal schedule for a workshop tour, the approach starts with a random population of possible schedules and, simulating a biological evolutions process tries to find better individual with respect to the criteria that wants to be optimized. To this aim, selection, crossover and mutation operators applied to the individuals are used to create the next generation. According to the Darwin laws, after a large number of generations a substantially better schedule will be obtained.

[1] nitahshah@gmail.com, [1]aakash3831@gmail.com [2]darshanabhogesara@gmail.com

2. LITERATURE REVIEW

Holland (1975) of the University of Michigan, Ann Arbor, conceived of these algorithms and published his important work (Holland, 1975) [1]. He provides a canonical example of a complex search through a space of ill-defined possibilities. He also proved a genetic algorithm which memorizes or arbitrarily creates introductory population at that time, genetic operators like selection, crossover and mutation, inside determined probabilities, are applied to deliver another generation that is considered better than its previous versions. The travelling salesman problem is also considered an optimization problem. It can be solved using a genetic algorithm since a genetic algorithm is used for solving optimization problems.

Mutluer et al. (2007) [2] proposed a Parameter Determination of Induction Machines by Hybrid Genetic Algorithms, and also proved that a genetic algorithm combined with other algorithms is well-known to be a powerful approach. In this paper, an efficient hybrid approach containing local search and genetic algorithms was presented. The purpose of using local search mechanisms is to provide better solution quality and to increase the convergence speed. Also, it was demonstrated that the performance of the proposed algorithms is significantly better than the conventional genetic algorithm methods.

Zurada (2010) [3] proposed Optimization Problems and Genetic Algorithms, and also presented an application of genetic algorithms (GAs) to a well-known travelling salesman problem (TSP), which is a challenging optimization task. Using the techniques of selection, crossover, and mutation borrowed from Darwin's evolution theory, GAs was able to find the optimal solution after generating only 24 populations of solutions instead of exploring more than a million possible solutions.

Chang, et al. (2001) [4] developed a new technique based on genetic algorithms (GA) that automatically determines, using a programmable goal function, a near-optimal allocation of resources and resulting schedule that satisfies a given task structure and resource pool.

Kaiafa et al. (2015) [5] developed an optimization method for multi-objective resource-constrained scheduling is developed which evaluates several resource-duration alternatives within each activity.

Nurdiawan et al. (2020) [6] developed an optimization method based on combining the Travelling Salesman Problem and Genetic Algorithm to optimize the total distance.

3. GENETIC ALGORITHM

The search for something better, an optimum, lies at the heart of evolution. A solution is defined as a point in a search space that contains all feasible solutions. EAs develop solutions to issues by considering a large number of potential options at once. The four principal classes of EAs are generally taken to be Genetic Algorithms (GAs), Evolution Strategies (ES), Evolutionary Programming (EP), and Genetic Programming. (GP).

GA is based on a stochastic process that will be inspired by nature. It is based on Darwin's evolution theory; it is one kind of search-based optimisation technique. It was first introduced by "John Holland" of the University of Michigan in the mid-1960s. He developed a computational technique that simulated the evolution process and applied it to Mathematical Programming. The GA revolves around the Genetic reproduction process and "Survival of the Fittest". It uses the concept of evaluating Biology (Natural Genetics & Natural Selection). Genetic algorithms mimic natural evolution by acting on a population to favour the creation of new individuals that 'perform' better than their predecessors, as evaluated using some criteria, such as an objective function. For more details, see [1]

3.1. Terminologies

Chromosomes: A chromosome (also sometimes called a genotype) is a set of parameters that define a proposed solution to the problem that the genetic algorithm is trying to solve. The set of all solutions is known as the population. Each iteration in the cycle produces a new generation of chromosomes.

Population: Population is the subset of all the possible solutions to the given problem.

Phenotype: Population in the actual real-world solution space.

Genotype: Population in the computational space.

Decoding: Transforming a solution from Genotype to Phenotype.

Coding: Transforming a solution from Phenotype to Genotype.

3.2. Basic structure:

The key idea of GA is to imitate the randomness of nature, where the natural selection process and behavior of the natural system enable the population of individuals to adapt to their surroundings. The solution created for this project is a genetic algorithm. Chromosomes are evaluated using a function. Selection can use several different selection algorithms, such as proportional selection and elitism. One-point, two-point, and random crossover algorithms are evaluated to determine their suitability to this problem domain. The mutation operator randomly increments and decrements integers within the chromosomes. An additional mutation operator is also discussed, which swaps two random genes within the chromosomes. Genetic algorithms are used to find multiple optimal solutions, see [1]. Figure 1 shows its scheme.

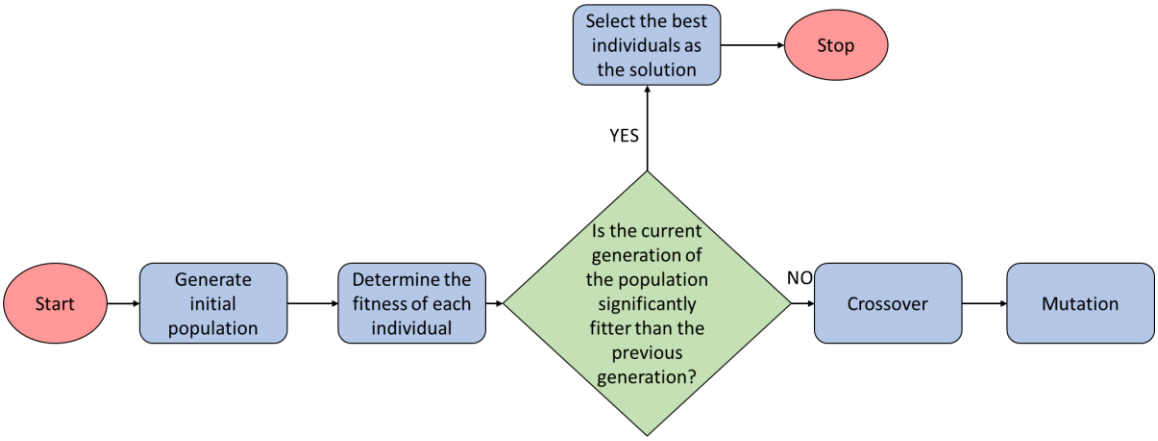

Figure 1

Here, first, we start with the initial population, select a set of solutions from the population, and calculate the fitness value for each value of individual. Then we fix the threshold value. If our individual value satisfies the threshold, then we find our best optimal solution; otherwise, select the individual with the best fitness value and go for crossover and mutation.

Selection:

Individual genomes are picked from a population for later breeding in the selection stage of a genetic algorithm (using the crossover operator).

A generic selection procedure may be implemented as follows:

1. For each individual, the fitness function is evaluated, producing fitness values that are then normalized. Normalization entails dividing each individual's fitness value by the sum of all fitness values, with the resultant fitness values equaling 1.
2. The accumulated normalized fitness values are computed: an individual's accumulated fitness value is the sum of its fitness value plus the fitness values of all preceding
3. Individuals; the accumulated fitness of the last individual should be 1 if the normalization step goes wrong.
4. A random number *R* between 0 and 1 is chosen.
5. The first individual whose accumulated normalized value is greater than or equal to R is chosen.

For many problems, the above algorithm might be computationally demanding. A simpler and a faster alternative uses the so-called stochastic acceptance.

There are five types of selection:

I. Roulette Wheel Selection

II. Rank Selection
III. Steady-State Selection
IV. Elitism Selection
V. Boltzmann Selection
VI. Tournament Selection

The selection approach is known as fitness proportionate or roulette-wheel selection if it is repeated until there are enough picked individuals. It's called stochastic universal sampling when, instead of a single pointer spinning numerous times, there are multiple, evenly spaced pointers on a wheel that is spun once. Tournament selection is the process of repeatedly selecting the best individual from a randomly chosen subset. Truncation selection is defined as taking the best half, third, or other proportion of the people.

Crossover:

Mitchell [1996] summarizes the concepts of selection, mutation, and crossover thus: "Selection increasingly focuses the search on subsets of the search space that feature schemas with observed above-average fitness, whereas crossover puts high-fitness 'building blocks' together on the same string to create strings of increasingly higher fitness."

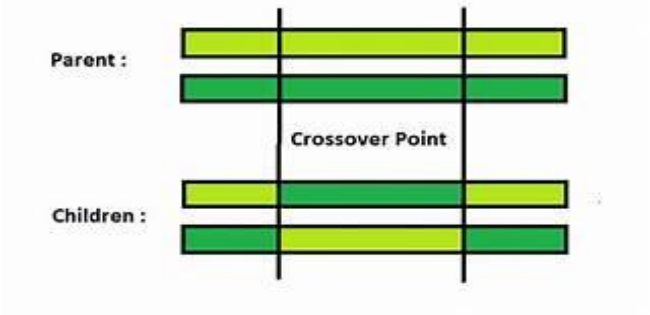

Figure 2. Scheme of Crossover.

In nature, both one-point and two-point crossover occur. Schemas that occur at either end of a chromosome will never be recombined in single-point crossover; the schema will be split up regardless of where crossover occurs, see Figure 2.

Mutation:

Schrödinger (1954) describes mutation as an isomeric change in the molecular structure of a gene. Once two of our electronic chromosomes have been chosen for reproduction, the algorithm arranges them side by side, element by element. A random number determines whether each element will be mutated, and if so, the element will be altered in some way. The mutation rate (or, more pedantically, the mutation probability per locus) is normally low, around 0.8 percent, yet it is crucial in terms of GA mechanics because:

- It expands the chromosome in previously unimaginable ways by expanding the search space.
- It helps prevent premature convergence

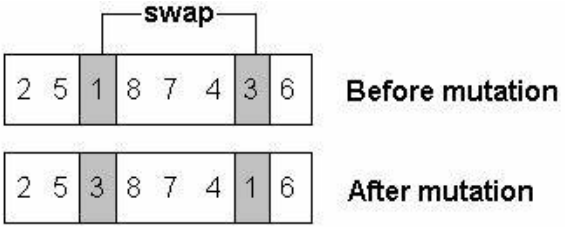

Figure 3: Scheme of Mutation.

4. METHODOLOGY:

The objective is to schedule the workshop tour in a certain period of D days minimizing the penalties of not fulfilling the clients' preferences and the costs. The closer the day is to the last day the better for the client.
The submission is scored according to the penalty cost for suboptimal scheduling. The penalization is scored as

$$\text{Penalty} = \sum_{g \in G} \sigma_g(d_g)$$

where d_g is the day assigned to group g and $\sigma_g(d)$ is a non-negative penalty function.
The constraints are the following:

- The total number of people attending the workshop each day must be between $L < U, L, U \in Z$. If even one day is outside these occupancy constraints, the submission will generate an error flag and will not be scored.
- Consolation gifts (of varying value) are given to clients according to their assigned day relative to their preferences. The preference cost is the sum of these expenses for each family. We assume there are p+1 choices, where Choice_0 is the case of *no consolation gifts and the other p the available options. Option I costs ci per person of the group*

The cost function is defined as a function of the number of visitors. Overcrowding implies for instance, extra cleaning costs, a drop of the sales, etc. This cost, in addition to the consolation gifts provided, is defined as

$$\text{Accounting penalty} = \sum_{d=D}^{1} \frac{(N_d - L)}{400} N_d \left(\frac{1}{2} + \frac{|N_d - N_{d+1}|}{50}\right)$$

where N_d is the occupancy of the current day, and N_{d+1} is the occupancy of the *previous* day.
For the initial condition $N_D = N_{D+1}$.

So, the objective function is given as

$$\text{Score} = \text{preference cost} + \text{accounting penalty}$$

The model is

Parameter

L: minimum of allowed visitors

U maximum of allowed visitors

$U(d, I, k)$: if day d is in rank k for family I , $k = 1, \dots q$

$U(d, I, q+1)$: if day d is not a preferred day for family I

$P(d, k)$: penalty of being assigned on day of ran $k, k = 1, \dots q+1$

N_i: number of members of family i

$Pr(j)$: price of gift j per person

$Dec(j)$: decrease of penalty of card gift

Variables: N_d: number of families going on day d

$x_{\{d,i\}}$: 1 if family i is assigned to go on day d

$y_{\{i,j\}}$: if family i get the gift card j

Example of preference cost:

$$\min \quad \sum_{k=1}^{q} \sum_{d=1}^{D} \sum_i x_{d,i} U(d,i,k) P(d,k) + \sum_i \sum_{j=1} \Pr(j) N_i y_{i,j} + \sum_{d=D}^{1} \frac{(N_d - L)}{400} N_d \left(\frac{1}{2} + \frac{|N_d - N_{d+1}|}{50}\right)$$

$$s.t. \qquad N_d \in [L, U], d = 1, \dots D$$

$$\sum_{d} x_{\{d,i\}} = 1, \quad i \text{ each group has a day}$$

$$\sum_{i} x_{d,i} N_i \in [L, U] \text{ each day the number of visitors are in the desired rank,}$$

$$\sum_{d} x_{d,i} U(d, I, 1) \leq y_{d,0} D, \text{ if group } i \text{ gets its favorite day gets no gift card}$$

$$\sum_{j} y_{i,j} = 1, \text{ one card is given for each group } j$$

5. EXAMPLE

We apply this methodology to a workshop tour before Christmas. The data are taking from the the Santa Workshop Tour Scheduling benchmark made publicly available through Kaggle, see [4]. In this sense we assume the option is open for $D = 100$ days there are 100 days $L = 125$, $U = 300, p = 10$ and the options are

Choice_1: one 50 Rs. gift card to Santa's Gift Shop

Choice_2: one 50 Rs. gift card, and 25% off Santa's Buffet (value 9 Rs.) for each family member

Choice_3: one 100 Rs. gift card, and 25% off Santa's Buffet (value 9 Rs.) for each family member

Choice_4: one 200 Rs. gift card, and 25% off Santa's Buffet (value 9 Rs.) for each family member

Choice_5: one 200 Rs. gift card, and 50% off Santa's Buffet (value 18 Rs.) for each family member

Choice_6: one 300 Rs. gift card, and 50% off Santa's Buffet (value 18 Rs.) for each family member

Choice_7: one 300 Rs. gift card, and a free Santa's Buffet (value 36 Rs.) for each family member

Choice_8: one 400 Rs. gift card, and a free Santa's Buffet (value 36 Rs.) for each family member

Choice_9: one 500 Rs. gift card, and free Santa's Buffet (value 36 Rs.) for each family member, and 50% off North Pole Helicopter Ride tickets (value $199) for each family member

Otherwise: one 500 Rs. gift card, and free Santa's Buffet (value 36 Rs.) for each family member, and free North Pole Helicopter Ride tickets (value 398 Rs.) for each family member

Data Description

Our task is to schedule the families to the workshop in a way that minimizes the penalty cost to him.

Each family has listed their top 10 preferences for the dates they'd like to attend the workshop tour. Dates are integer values representing the days *before* Christmas, e.g., the value 1 represents Dec 24, the value 2 represents Dec 23, etc. Each family also has several people attending. $n\_$ People. Every family must be scheduled for one and only one assigned_day.

So, here we use two data sets: (1): Family_data (here we show the results and data for the first 11 families)

family_id	choice_0	choice_1	choice_2	choice_3	choice_4	choice_5	choice_6	choice_7	choice_8	choice_9	n_people
0	52	38	12	82	33	75	64	76	10	28	4
1	26	4	82	5	11	47	38	6	66	61	4
2	100	54	25	12	27	82	10	89	80	33	3
3	2	95	1	96	32	6	40	31	9	59	2
4	53	1	47	93	26	3	46	16	42	39	4

5	32	59	12	3	60	26	35	50	5	2	4
6	88	4	1	3	91	32	39	57	28	99	2
7	25	11	52	48	10	17	88	50	95	66	5
8	18	60	1	12	89	33	16	10	53	67	4
9	1	88	39	50	26	18	96	47	46	28	7
10	96	92	8	5	67	12	57	34	80	46	7

Table 1: Family data.

family_id	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Assigned day	52	26	100	2	53	32	88	25	18	50	92	19	98	54	45

Table 2 Sample Submission.

6. CONCLUSION:

This study is focused on the application of Genetic Algorithms (GA) to the Workshop Tour Scheduling Problem, a complex combinatorial optimization task involving the allocation of workshop visits to minimize the penalty cost and time. To schedule the families to the workshop in such a way that minimizes the penalty cost to him, we consider the family data, preferences to design and implement a GA-based approach. We demonstrated its effectiveness in generating near-optimal solutions with suitable computational time. The data is taken from Kaggle, and by applying the GA algorithm, we achieved near-optimal solutions with 19 epochs. Integrating real-time constraints or families' preferences into the scheduling model could improve its applicability in practical scenarios.
For workshop tour scheduling, the output for the minimum penalty cost for each family is as follows:

0 Min on epoch: 80304.25928664337
1 Min on epoch: 78997.98998730004
2 Min on epoch: 79280.38469785672
3 Min on epoch: 79161.3894790741
4 Min on epoch: 79161.3894790741
5 Min on epoch: 80638.53043679858
6 Min on epoch: 80114.10901887505
7 Min on epoch: 80114.10901887505
8 Min on epoch: 79870.66489397602
9 Min on epoch: 79870.66489397602
10 Min on epoch: 80249.98132391725
11 Min on epoch: 81318.33646736816
12 Min on epoch: 80800.45181871949
13 Min on epoch: 81351.6534671493
14 Min on epoch: 82687.81478456593
15 Min on epoch: 82433.71650093321
16 Min on epoch: 83195.78575654392
17 Min on epoch: 83201.97267744257
18 Min on epoch: 83210.99861710054
19 Min on epoch: 83209.37219948944

RECEIVED: SEPTEMBER, 2024.
REVISED: FEBRUARY, 2026.

REFERENCE

1. CHANG, C. K., CHRISTENSEN, M. J., & ZHANG, T. (2001). Genetic algorithms for project management. **Annals of Software Engineering,** *11*(1), 107-139.
2. DAVID EDWARD, G. (2002). **Genetic Algorithms in Search, Optimization, and Machine Learning.** Addison-Wesley.
3. HOLLAND, J. H. (1987). **Genetic algorithms and classifier systems: foundations and future directions** (No. LA-UR-87-1863; CONF-870775-1). Univ. of Michigan, Ann Arbor, MI (United States).
4. KAGGLE, **Santa's workshop Tour**: Santa's Workshop Tour 2019 | Kaggle

5. KAIAFA, S., & CHASSIAKOS, A. P. (2015). A genetic algorithm for optimal resource-driven project scheduling. **Procedia Engineering**, *123*, 260-267.
6. MUTLUER, M., BILGIN, O., & ÇUNKAŞ, M. (2007). Parameter determination of induction machines by hybrid genetic algorithms. In **Knowledge-Based Intelligent Information and Engineering Systems: 11th International Conference, KES 2007, XVII Italian Workshop on Neural Networks,** *Vietri sul Mare, Italy, September 12-14, 2007. Proceedings, Part I 11* (pp. 116-124). Springer Berlin Heidelberg.
7. NURDIAWAN, O., PRATAMA, F. A., KURNIA, D. A., & RAHANINGSIH, N. (2020, March). Optimization of travelling salesman problem on scheduling tour packages using genetic algorithms. In **Journal of Physics: Conference Series** (Vol. 1477, No. 5, p. 052037). IOP Publishing.
8. ZURADA, J. (2010). Optimisation Problems and Genetic Algorithms. **University of Louisville, USA. Review of Business Information System,** *14*.