A Genetic Algorithm With Composite Chromosome for Shift Assignment of Part-time Employees

Ning Xue
ASAP Research Group
School of Computer Science
The University of Nottingham, UK
Ning.Xue1@nottingham.ac.uk

Dario Landa-Silva
ASAP Research Group
School of Computer Science
The University of Nottingham, UK
dario.landasilva@nottingham.ac.uk

Isaac Triguero
ASAP Research Group
School of Computer Science
The University of Nottingham, UK
Isaac.Triguero@nottingham.ac.uk

Grazziela P. Figueredo
IMA Research Group
School of Computer Science
The University of Nottingham, UK
Grazziela.Figueredo@nottingham.ac.uk

***Abstract*—Personnel scheduling problems involve multiple tasks, including assigning shifts to workers. The purpose is usually to satisfy objectives and constraints arising from management, labour unions and employee preferences. The shift assignment problem is usually highly constrained and difficult to solve. The problem can be further complicated (i) if workers have mixed skills; (ii) if the start/end times of shifts are flexible; and (iii) if multiple criteria are considered when evaluating the quality of the assignment. This paper proposes a genetic algorithm using composite chromosome encoding to tackle the shift assignment problem that typically arises in retail stores, where most employees work part-time, have mixed-skills and require flexible shifts. Experiments on a number of problem instances extracted from a real-world retail store, show the effectiveness of the proposed approach in finding good-quality solutions. The computational results presented here also include a comparison with results obtained by formulating the problem as a mixed-integer linear programming model and then solving it with a commercial solver. Results show that the proposed genetic algorithm exhibits an effective and efficient performance in solving this difficult optimisation problem.**

***Index Terms*—personnel scheduling, shift assignment, genetic algorithms, multiple objectives, multi-skills, flexible shift length**

I. Introduction

The typical process of personnel scheduling consists of several stages. Tien and Kamiyama [1] decomposed the problem into five separate but related stages, while Ernst et al. [2] divided it into six phases. If the demand is time-based (e.g. 3 workers are required on Tuesday 7am-8am) and the worker assignment is shift-based (e.g. worker A is on a morning shift from 7am to 11am), then these stages can be broadly classified into three main ones as follows. The first stage, *demand modelling* determines how many workers are required at different times over a planning horizon, which is typically of one week. The second stage, the *shift designing* involves designing a set of shifts and calculating their required workers to meet the demand. The third stage, *shift assignment* is to allocate workers to each individual shift, while satisfying various constraints (e.g. availability, preference and labour law) in order to achieve certain objective(s) such as minimising labour cost and maintaining fairness in the assigned workload.

Demand modelling can usually be tackled as a naturally separate phase [1]. There is a variety of solution methods to solve the shift design and shift assignment stages, but there are two main approaches. Firstly, both shift design and assignment can be solved in one phase where a mathematical model is formulated and solved with a suitable solution method [15] (e.g. by a mathematical programming solver). Theoretically, this approach can provide an overall optimal solution to the combined problem. The complexity and problem size depend on a number of factors. For example, the flexibility of shift patterns (i.e. flexible start time and end time) can result in a very large problem, which is hard to solve efficiently. For this reason, exact methods of decomposition and problem-size reduction [16] [17] as well as heuristic methods [4] [5] [6] [10] have been proposed to tackle the combined problem. Secondly, shift design and shift assignment can be solved as two independent problems [7] [8] [11]. This can make the problems easier to tackle. However, an overall optimal solution for shift assignment, given an optimal solution for shift design, cannot be guaranteed. Studies about the standalone shift design problem can be found in [9], [12]–[14].

This paper addresses the shift assignment problem, where for each shift the following information is given: shift lengths, their start/end times, the number of required workers and the required skills or roles. Our challenge is to create weekly schedules for a retail store with up to 50 workers. In this problem, the term skill and role are interchangeable, for example, a worker on a cleaning role or with cleaning skills refer to the same requirement. In addition, in order to meet the demand on the number of workers and skills for each shift, the generated schedules have to take into account worker availability, worker preferences (e.g. not to work with someone, preferred working hours) and regional labour laws. The objectives are to minimise total labour cost, while maintaining

a fair distribution of working hours and unpopular shifts. The problem has the following characteristics (detailed description is given in section II):

- Workers are paid according to their age and seniority.
- Each worker usually has multiple skills (roles), e.g. cleaner, counter, etc.
- The total number of shifts in a week can be over 100 and the total number of workers can be up to 50.
- The majority of workers are part-time, resulting in great variability in their availability and preferences.
- The start time of a shift is flexible and the shift length varies from 3 to 12 hours. For example, a shift can be 6.30am to 10am, 6.30am to 11am, 8am to 2pm and so on.
- The given shifts (i.e. shift design stage which is not considered here) were constructed considering working regulations and practices (e.g. forbidden sequence of shifts and incompatibility between workers).

A genetic algorithm (GA) with a composite chromosome representation is proposed in order to tackle the shift assignment problem described above, which involves mixed-skills, flexible shifts and multiple objectives. GAs have been successfully applied to several personnel scheduling problems, such as nurse scheduling [18]–[21], [37] and doctor rostering [39], [42], job rotation in manufacture industry [36], transportation [38], [40], [41], service centres [43], and project scheduling [44]. In general, the encoding scheme in these GAs is either a permutation of the workers to be scheduled or a permutation of the shifts/tasks to be covered. A distinctive feature of the GA design described in this paper is the solution encoding in the form of a composite chromosome that contains the day schedules, the shifts and also the workers suitable for each individual shift.

The standalone shift assignment problem is usually highly constrained and difficult to solve. It shares some similarities with the nurse scheduling problem, which is a variant of the personnel scheduling problem, and focuses on assigning shift types to nurses in each day of the planning period. The nurse rostering problem has been shown to be NP-complete [24]. Moreover, when multiple criteria and mixed-skills are involved, the problem becomes very complex to solve [23]. The demand in nurse rostering problem is shift-based and usually obtained directly from a specification of the number of nurses required for each shift of the planning period [2]. In addition, the start and end times of a shift in nurse rostering are fixed. For example, there are only a few shift types like morning shift (6am to 12pm), afternoon shift (12pm to 6pm) and night shift (6pm to 12am). As mentioned above, the problem tackled in this paper involves flexible shifts in terms of their length and start/end times. This difference prevent us from directly applying the techniques (e.g. [20], [21]) used for nurse rostering to our problem. This is because such flexibility in the shifts can result in a much larger size of "shift patterns" [20] especially if workers' available times are highly variable. The problem tackled here is also different from other personnel scheduling problems involving flexible working patterns (e.g. [10], [22]). Although there are some similarities in the constraints, the heuristics that have been developed to tackle those other problems are tailored to take full advantage of the specific problem. Therefore, the problem tackled in this paper cannot be solved easily by "off-the-shelf" approaches from the literature.

The GA method proposed in this paper is tested on a number of problem instances with different features and sizes. Results from a comparison with a commercial solver indicate that the proposed GA is able to obtain high-quality solutions in less computation time. The remainder of this paper is organised as follows: the problem and data instances are described in detail in Section II; the constraint handling and penalty functions are illustrated in Section III; the solution method is presented in Section IV; followed by the computational experiments in Section V. Conclusions are drawn in the last section.

II. Problem Description and Test Instances

In this section, we first describe the structure of the problem instances, followed by the problem description. We also refer to relevant parts of the data employed in a problem instance.

A. Problem Instance Data

A set of problem instances reflecting characteristics of the real-world scenarios was generated using historical data from a local retail store. The data for these problem instances is available at: https://assets.pxtech.com/misc/schedulingproblem/WCCI2018Instances.zip. The instances are classified into small, medium and large size. Five instances of each size were generated. The instances groups are called I1, I4 and I8, where the number corresponds to the number of job skills (roles) that are in demand. For example, I4 are the instances where 4 different skills or roles are required when generating the schedule. The data contained in each problem instance is listed in Table I.

TABLE I
Data contained in an instance of the shift assignment problem

Store profile	Max shift length
	Night shift end after
	Morning shift start before
	Max consecutive working days
	Incompatible worker set
Worker profile	Hourly Payment
	Day start time
	Day end time
	Max working hours in a week
	Max working hours in a day
	Max working days in a week
	Days on and off
Demand	Weekday
	Shift start time
	Shift end time
	Required role

B. Problem Description

In this paper, we follow the terminology from [25] to define shift, tour and schedule. Some data from problem instance I4-1 is used next to illustrate the features of the problem.

A *shift* is specified by the follwing: day of week, required role (skills), start and end times. The length of a shift ranges from 3 to 12 hours. An example of shift demand is presented in Table II. A *tour* is composed by a sequence of 'days on' and 'days off', where each day on consists of one shift. A *schedule* is a set of tours covering the length of the planning horizon, which in the case considered in this paper is of one week.

TABLE II
EXAMPLE OF SHIFT DEMAND WITH EACH COLUMN SHOWING ONE SHIFT

Day of week	Monday	Monday	Monday
Shift start	8	8	11.5
Shift end	12	13.5	15.5
Required role	2	1	0

A solution to an instance of the shift assignment problem is to assign a tour to each worker. When constructing a workforce schedule, the primary objective is to provide the staffing required for each shift, while minimising labour cost. The cost of a shift is calculated by the shift length (in hours) multiplied by the hourly payment for the worker assigned to the shift. The secondary objective is to fairly distribute *unpopular shifts* among workers. An unpopular shift is undesirable for most workers. For our case, those shifts starting before 12pm or ending after 8pm are identified as unpopular. The third objective is to ensure that each worker is assigned a similar total number of working hours across the planning period.

Any feasible solution should cover all the requirements by official regulations and satisfy the availability and preference of workers, that is:

- Workers availability: in the problem instance, availability is expressed by (i) the start and end times that a worker is available in a working day (see Table III); as well as by (ii) an indication of whether a worker is available for duty or not for each day of the week (see Table IV).
- The maximum working hours in a week (see Table III).
- The maximum working hours in a day (see Table III).
- The maximum working days in a week (see Table III).
- The maximum consecutive working days allowed in a planning horizon (see Table III).

TABLE III
WORKER INFORMATION

Worker ID	0	1	2
Hourly payment	7.7	7.7	5.6
Start time	6	6	6
End time	23	15	23
Max consecutive working day	5	5	5
Max working hours in a week	50	40	40
Max working hours in a day	12	12	12
Max working days in a week	5	7	7

The store manager considers constraints related to employee satisfaction and working efficiency, that is:

- Workers qualifications: whether a worker has the required skills (roles) for a given shift. A worker can possess multiple skills, e.g. for counter and cleaner roles (see Table V).
- Sequence constraints: a worker cannot be assigned to a shift that starts before 12pm on day k immediately after a shift ending after 8pm on day $k-1$. This is to ensure enough rest time between shifts.
- Incompatible worker constraints: workers that do not get along with each other will not be assigned to the same shift. Similarly, workers that are too familiar with each other will not be assigned to work together. This is to prevent possible collusion behaviour, which has a negative impact on sales. For example, in problem instance I4-1, workers 5 and 6 are identified as mutually incompatible.

TABLE IV
WORKER DAYS ON

Worker ID	0	1	2
Monday	1	1	1
Tuesday	0	1	1
Wednesday	1	1	1
Thursday	1	0	1
Friday	1	1	1
Saturday	1	1	1
Sunday	1	1	1

TABLE V
WORKER ROLES

Worker ID	0	1	2
Role 0	1	1	1
Role 1	0	1	1
Role 2	0	1	1
Role 3	1	1	1

The constraints mentioned above are all treated as hard constraints in this problem.

III. CONSTRAINT HANDLING AND PENALTY FUNCTIONS

In this section we present the mathematical formulation for the shift assignment problem considered in this paper. The terminology is introduced first, followed by the formulation of constraints and the objective function.

Decision Variables

- $x_{jk}^{i}: \begin{cases} 1 & \text{if worker } i \text{ is assigned to shift } j \text{ on day } k. \\ 0 & \text{otherwise.} \end{cases}$
- $z_{k}^{i}: \begin{cases} 1 & \text{if worker } i \text{ is assigned to a shift on day } k. \\ 0 & \text{otherwise.} \end{cases}$
- b_i: number of unpopular shifts assigned to worker i.
- b_{avg}: average number of unpopular shifts assigned considering all workers.
- t_i: number of working hours assigned to worker i.
- t_{avg}: average working time assigned considering all workers.

Parameters

- p_b: penalty cost due to the difference of one shift to b_{avg}.
- p_t: penalty cost due to the difference in one hour to t_{avg}.
- W_n: weight associated to the n^{th} component of the penalty function P_n.

Data Constants

- α_{tjk} : $\begin{cases} 1 & \text{if time period } t \text{ is covered by shift } j \text{ in day k.} \\ 0 & \text{otherwise.} \end{cases}$
- c^i_{jk}: cost of assigning worker i to shift j of day k.
- l_{jk}: length of shift j on day k.
- l^i_{max}: maximum number of working hours allowed for worker i in a week.
- w^i_{max}: maximum number of working days per week allowed for worker i.
- w^i_{cons}: maximum number of consecutive working days per week allowed for worker i.
- h^i_{max}: maximum number of working hours per day allowed for worker i.

Data Sets

K: set of consecutive days in a planning horizon, $k \in K$.

I: set of workers, $i \in I$.

T: set of time periods of equal length, $t \in T$.

S_I: set of subsets S'_I of incompatible workers.

S'_I: set of incompatible workers, $S'_I \subset S_I, S_I \subset I$.

J^i_k: set of shifts for which worker i is available in day k.

N^i_k: set of night shifts (ending after 8pm) in day k that can be assigned to worker i, $N^i_k \subset J^i_k$.

F^i_{k+1}: set of morning shifts (starting before 12pm) in day $k+1$ that are forbidden for worker i given that the worker is assigned shift $j \in N^i_k$, $F^i_{k+1} \subset J^i_{k+1}$.

Constraint Handling

In general, there are three widely used constraint handling methods in a GA. The first approach implements a specialised encoding scheme and operators, so that the search takes place over only on feasible regions. The second method repairs the solution if the genetic operators lead to infeasibility. Thirdly, penalties to the objective function are adopted to penalise constraint violations and therefore to reduce the fitness of unfeasible solutions. Additionally, other constraint handling techniques can be found in [28]–[31].

Our problem has constraints related to (i) worker's working hours (i.e. maximum working hours in a day, maximum working hours in a week); (ii) working days (i.e. maximum working days in a week); and (iii) workers compatibility and shift sequences. All these constraints limit the efficacy of repair operators, as changing any shift assignment will likely affect other assignments in the schedule. To overcome this limitation, the implementation of specialised encoding and genetic operators (first approach mentioned above) coupled with the application of penalties (third approach mentioned above) are adopted here. The mutation and crossover operations are implemented in a way that a worker cannot be assigned to more than one shift per day. This avoids, for instance, workers with lower wages being assigned to the majority of shifts. The encoding scheme implemented here (Section IV) automatically handles those constraints related to worker availability and skill sets. The rest of the constraints are handled by the penalty approach in the objective function. The related notations and fitness function are given below:

Objective and Penalty Functions

The labour cost is calculated by: $O =$

$$\sum_{i \in I} \sum_{k \in K} \sum_{j \in J^i_k} c^i_{jk} x^i_{jk} \tag{1}$$

The penalty for the assignment of unpopular shifts is given by: $P_1 =$

$$\sum_{i \in I} p_b \left| b_i - b_{avg} \right| \tag{2}$$

The penalty for unfair assignment of working ours is given by: $P_2 =$

$$\sum_{i \in I} p_t \left| t_i - t_{avg} \right| \tag{3}$$

The penalty for exceeding the maximum number of working hours in a week is given by: $P_3 =$

$$\sum_{i \in I} \Big\{ max \Big\{ 0, \sum_{k \in K} \sum_{j \in J^i_k} x^i_{jk} l_{jk} - l^i_{max} \Big\} \Big\} \tag{4}$$

The penalty for exceeding the maximum number of working hours in a day is given by:$P_4 =$

$$\sum_{k \in K} \sum_{i \in I} \Big\{ max \Big\{ 0, \sum_{j \in J^i_k} x^i_{jk} l_{jk} - h^i_{max} z^i_k \Big\} \Big\} \tag{5}$$

The penalty for exceeding the maximum number of working days in a week is given by:$P_5 =$

$$\sum_{i \in I} \Big\{ max \Big\{ 0, \sum_{k \in K} z^i_k - w^i_{max} \Big\} \Big\} \tag{6}$$

The penalty for exceeding the maximum number of consecutive working days in a week is given by: $P_6 =$

$$\sum_{i \in I} \Big\{ max \Big\{ 0, w^i_{cons} - \sum_{k=k'}^{k'+w^i_{cons}} z^i_k \Big\}, k' = 1, 2, ..., |K| - w^i_{cons} \Big\} \tag{7}$$

The penalty for violating worker incompatibility is given by: $P_7 =$

$$\sum_{k \in K} \sum_{t \in T} \Big\{ max \Big\{ 0, \sum_{i \in S'_I} \sum_{j \in J^i_k} x^i_{jk} \alpha_{tjk} - 1 \Big\} \Big\} \tag{8}$$

The penalty for invalid shift sequences (night followed by morning) is given by: $P_8 =$

$$\sum_{i \in I} \Big\{ max \Big\{ 0, \sum_{j \in N^i_k} x^i_{jk} + \sum_{j' \in F^i_{k+1}} x^i_{j'k+1} - 1 \Big\}, k = 1, 2, ..., |K| - 1 \Big\} \tag{9}$$

The overall objective function is to minimise the weighted sum of the above:

$$obj = \min(O + \sum_{n=1}^{8} W_n P_n) \tag{10}$$

IV. Proposed genetic algorithm

Our solution approach was inspired by previous successful applications of GAs on nurse rostering problems [20], [21], [27]. It differs, however, from nurse rostering solutions, as our chromosome is not constructed by genes representing indexes of "shift patterns". Instead, a composite chromosome is used to encode the various aspects of a solution in an effective manner; it also facilitates the application of the genetic operators.

The solution is encoded as a composite chromosome containing seven vectors (i.e. sub-chromosomes). It also varies from one day to another due to the typical variations in demand in the retail store. Each vector therefore represents the shift assignment in a day of the week. The location of a gene represents a shift and its value indicates the worker assigned to the shift. Each gene is associated with an allele set containing all workers that are available and their skill sets to be assigned to the shift.

Fig.1 shows an example of a composite chromosome constructed with 7 sub-chromosomes of index k. The 7th sub-chromosome contains the 5 required shifts in that day. The allele set of the first shift (shift1) contains 5 workers who are available and able to work on shift1. In the example, x^1_{17}=1 stands for worker 1 (i=1) is assigned to shift 1 (j=1) on day 7 (k=7). Each sub-chromosome behaves differently and is not affected by genetic operations on the other sub-chromosomes. The mutation operator assigns a shift to a randomly selected worker from its corresponding allele set. There are three types of crossover [35] implemented here: single point, two point and uniform crossovers. The mutation and crossover functions are implemented to ensure the constraints regarding shifts assigned to no more than one worker are met in each sub-chromosome. Fig.2 shows an example of the one point crossover operator.

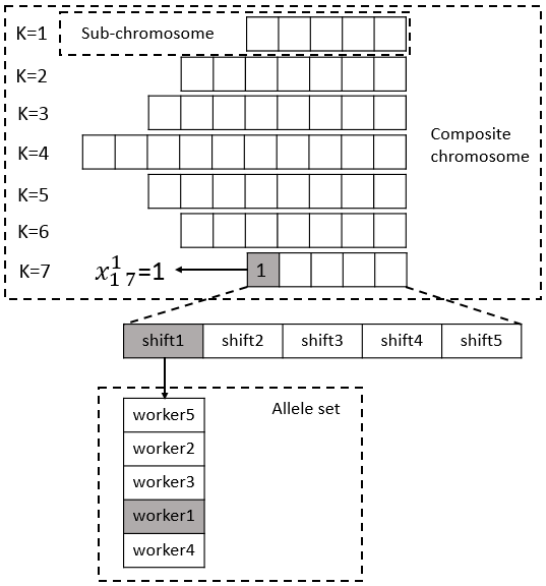

Fig. 1. Example of composite chromosome

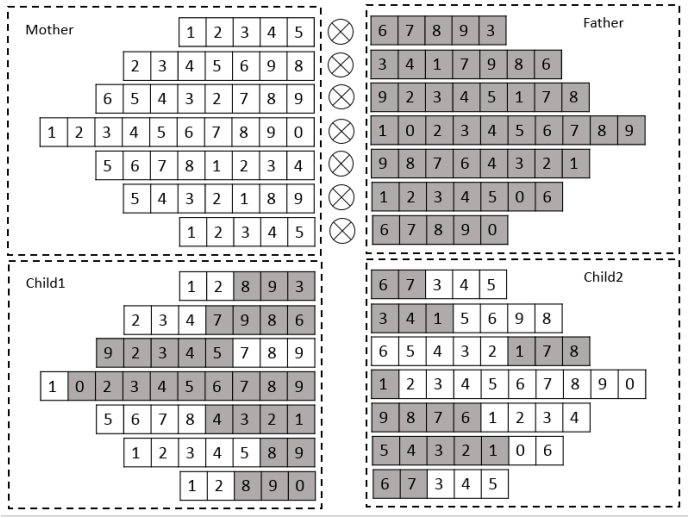

Fig. 2. Example of one point crossover

This composite representation is chosen as it automatically handles constraints related to worker availability and ability. Those highly restrictive constraints are usually difficult to handle with a general simple representation. Preliminary experiments conducted in our work show that a two-dimensional binary representation (i.e. with the gene representing whether a worker appears on a shift or not) and a one-dimensional array (i.e. with each gene representing a worker and its value being the "shift pattern" index, as implemented in [20]) are less effective for this problem. Another reason for using a composite chromosome instead of a one-dimensional array is that there could be more than 100 shifts to be scheduled in a week. This would require a very long one-dimensional chromosome to represent a weekly schedule. Some initial tests demonstrate a very slow convergence rate as the power of some genetic operators (e.g. single and two point crossover) is reduced. This is due to the fact that the shift assignment might remain unchanged after several generations. Although this issue can be addressed by uniform crossovers, we believe that high-quality schedules are evolved more efficiently through single and two point crossovers. As observed in [26], [27], the success of a GA is usually attributed to the validity of the building block hypothesis, which relies on the crossover operator being able to combine good partial solutions (i.e. building blocks) into complete good solutions. Therefore, in addition to uniform crossover, single point and two point crossovers are also implemented and evaluated here.

V. Experiments and Results

A. Algorithm Tuning Experiments

The proposed GA with composite chromosome was implemented in C++ and run on a PC with Intel i7 2.40GHZ processor and 4GB RAM, which is similar to the retail store's equipment. The parameters and penalty weights are given in Tables VI and VII. The values of W_1 and W_2 (i.e. penalty weights for unfair assignment of unpopular shifts and unfair

assignment of total working hours respectively) were set in consultation with the store manager. The other weight values were set based on some preliminary experiments over a few instances, on the understanding that they may not necessarily be the best penalty weights for the problem.

TABLE VI
PARAMETERS FOR THE GENETIC ALGORITHM

Parameters	Settings
Population size	10
Crossover	One point crossover
Mutation rate	0.01
Stop criterion	1000 non-improving generations
Selection	Elitism

TABLE VII
PENALTY WEIGHTS FOR OBJECTIVE FUNCTION

Weight	Value
W_1	10
W_2	5
W_3	50
W_4	50
W_5	50
W_6	100
W_7	100
W_8	100

Preliminary experimentation revealed that a small population size evolved over a large number of generations produces good solutions in short computation time. The Irace package [45] was applied to help parameter tuning. Population parameters were set by categorical values within (5, 10, 20, 30, 40, 50), while mutation parameters were real values within range (0.01, 0.5). The max number of experiments was set as large as 1000; other parameters for Irace were set to default values. The results suggested that population size of 10 or 30 and mutation rate of 0.01 are the best suited. Some experiment results corresponding to the average objective values obtained with different settings for mutation rate, population size and crossover operator are given in Table VIII. These results suggest that the parameters $mutationrate = 0.01$, $population = 10$ and one point crossover operator are more suitable in terms of both efficiency and solution quality. Although $populationsize = 30$ and one point crossover lead to the smallest penalty cost, it required much longer computing time. These results are based on 20 independent runs. It can also be clearly seen that both one and two point crossovers perform better than the uniform crossover, confirming the building block hypothesis [26] [27] for this problem.

In order to illustrate the effect of the composite chromosome encoding scheme, results on instance I1-1 are compared to results obtained when using a two-dimensional array for chromosome encoding instead. As mentioned in Section IV, in a two-dimensional array chromosome: 1) the first dimension is associated to shifts and the second dimension is associated to workers; 2) each gene is a Boolean value indicating whether a worker is assigned to the given shift or not. The algorithm was executed with each of the two chromosome encodings using the same initial solution as starting point. Parameter values and penalty weights were as shown in Tables VI and VII. Results were obtained over 5 independent runs. Results of this experiment are shown in Fig.1 corresponding to the average results for problem instance I1-1. It is clear from the graph that when using the composite chromosome proposed in the paper, the GA converges faster than when using the two-dimensional chromosome.

TABLE VIII
COMPARISON OF RESULTS OBTAINED WITH DIFFERENT SETTINGS FOR MUTATION RATE, POPULATION SIZE AND CROSSOVER OPERATOR

	Mutation	Population	Penalty cost	Time(s)
Uniform crossover	0.01	30	2218.9	344.8
	0.01	10	2292.8	118.5
	0.02	30	2276.7	336.2
	0.02	10	2232.1	114.7
	0.05	30	2340.7	391.9
	0.05	10	2320.9	157
Two point crossover	0.01	30	2206.7	158.6
	0.01	10	2234.6	72.7
	0.02	30	2238.7	147.3
	0.02	10	2251.8	161.5
	0.05	30	2264.3	75.2
	0.05	10	2290.7	86.1
One point crossover	0.01	30	2189.4	155.3
	0.01	10	2222.6	75.7
	0.02	30	2231.5	157.8
	0.02	10	2236.6	65.9
	0.05	30	2254.5	142.6
	0.05	10	2267.7	69.9

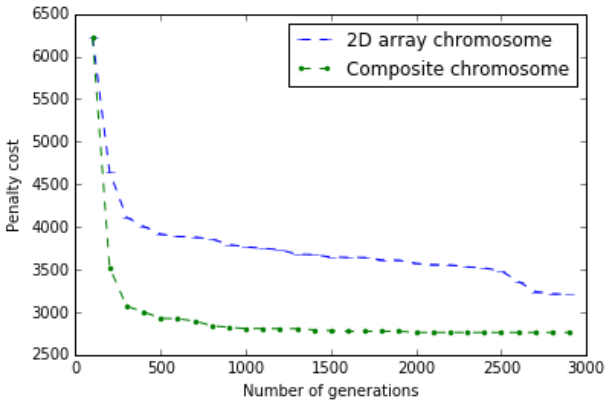

Fig. 3. Results with different encoding schemes on instance I1-1

B. Algorithm Performance Experiments

In order to analyse the performance of the proposed solution method, we tested the algorithm on 15 problem instances (described in section II) using the parameter values and penalty weights given in Tables VI and VII. In order to obtain statistically sound results, all experiments were conducted considering 20 independent runs (mean values were recorded) over all 15 data sets. For each instance, all runs were started with the same initial solution created at random (Section V-A). The experiments were also run on a PC with Intel i7 2.40GHZ processor and 4GB RAM.

A comparison is also made between results obtained with the GA and results obtained by solving the instances using a mixed-integer linear programming (MILP) model [32] (*LP approach*). The objective of the LP approach is to minimise (1)+(2)+(3) while the penalty functions (4) to (9) were transformed into linear inequalities as constraints. All the 15 problem instances were solved to feasibility by the LP approach. The values for penalty weights W_1 and W_2 were the same for both the GA and the LP approach. An open source solver (Lpsolve 5.5 [33]) was not able to return any solution within 12 hours. We then applied the commercial solver Gurobi 7.5 [34] using the default settings as the LP solver. Results of this comparison are shown in Table IX.

Table IX shows that the GA consistently obtains feasible solutions. The *Average* and *Time(s)* in the *GA* columns are the average objective values and the average running time, respectively. The *Result* in the *LP* column is the objective function value of the integer feasible solution obtained by the Gurobi solver at the same computation time as the GA approach for the same problem instance. *LB* (lower bound) is the best known lower bound obtained at 600s by the Gurobi solver. The solver was stopped at 600s because memory would normally run out after that time. The same results as in Table IX are shown in Fig.4 for better visualisation.

From these results, it can be seen that most instances were solved within 3 minutes. In general, the GA obtained comparable solutions to those by the LP approach in the same amount of running time for the small and medium-sized problem instances (i.e. I1 and I4). The GA exhibits advantage in solving the large instances more efficiently. We believe that this is due to the increased search space that the LP approach faces as the number of job skills increases. The solver requires far more computational time due to the difficulty of covering the demand of multi-skills. This issue however has far less influence on the GA. In fact, thanks to the composite chromosome encoding, the search space to be explored by the GA is reduced, as the size of a shift's allele set is limited by the restriction of specific skills for that shift.

Due to the high costs of a powerful commercial LP solver such as Gurobi, this is not a practical solution for many retail stores. The solutions and running times of the GA approach are acceptable to the store manager as they are not only feasible but also comparable to the LP approach and close to the lower bound. Therefore, we suggest that the proposed GA is an efficient and economic solution method for the shift assignment problem studied in this paper.

TABLE IX
COMPARISON WITH LINEAR PROGRAMMING APPROACH

		GA		LP	
Index	Instance	Average	Time(s)	Result	LB
1	I1_1	2790.12	141.16	2836.45	2668.24
2	I1_2	2955.05	98.30	2890.35	2799.15
3	I1_3	2525.74	81.03	2494.55	2421.30
4	I1_4	2567.89	104.51	2568.09	2466.67
5	I1_5	3243.68	145.42	3221.74	3086.82
6	I4_1	2288.41	57.49	2321.91	2126.82
7	I4_2	2300.21	146.51	2275.76	2183.33
8	I4_3	2295.91	153.04	2287.09	2090.48
9	I4_4	2668.36	170.78	2760.51	2475.82
10	I4_5	2686.67	155.08	2661.05	2494.88
11	I8_1	3053.93	131.88	3212.04	2899.55
12	I8_2	2841.90	169.96	2983.40	2615.41
13	I8_3	2838.79	170.59	2832.86	2683.58
14	I8_4	2715.20	160.89	2715.72	2478.89
15	I8_5	2751.91	176.43	2897.46	2630.99

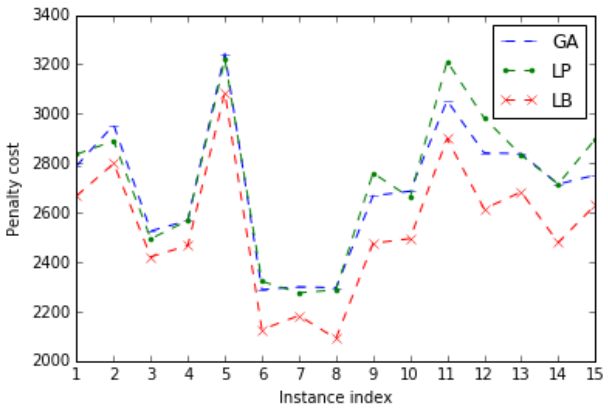

Fig. 4. Visualisation of comparison with linear programming approach

VI. CONCLUSIONS AND FUTURE WORK

Workforce scheduling is one of the most investigated optimisation topics in the literature. However, the variety in the type and size of this problem makes it difficult to identify a solution approach that works well across scenarios. In this paper, we have tackled a shift assignment problem arising in many retail stores and other places in the services industry. The problem consists of assigning workers to shifts in each day of the week. Multiple skills and shift flexibility arise in the problem. That is, workers have multiple skills, which allow them to take on different roles. Also, the length and start/end times of shifts is flexible, giving rise to a large number of shift patterns over the planning period. In addition, the quality of the weekly schedule is measured using multiple criteria.

We have proposed a GA that uses an effective composite chromosome encoding scheme. Experimental results on 15 problem instances have demonstrated the capability and consistency of this solution method in solving different problem instances. Compared to formulating and solving this problem as a MILP model using a powerful commercial solver, the GA approach demonstrated better performance with regards to efficiency in solving large sized problems (i.e. 8 work skills involved in the shift demand). The contribution of this paper is the design of a robust GA for the shift assignment problem and the creation of a set of problem instances, which are now publicly available. The algorithm itself is also relatively easy to implement and provides potential for its application to other similar optimisation problems.

We outline opportunities for future work. Even though the current solution quality is acceptable by the store manager, there is still a gap to the lower bound and hence the possibility of even better solutions to be found. Also, it will

be interesting to further test the capabilities of the current composite chromosome encoding, as complexity is added to the problem. The most challenging constraints in this problem are worker availability and ability. These are handled well by our encoding scheme. However, additional objectives and constraints are being considered by the store manager. Further tests and possible improvements to the algorithm are therefore required to handle highly-constrained cases and to deal with feasibility issues in the future.

REFERENCES

[1] Tien, J.M. and Kamiyama, A., 1982. "On manpower scheduling algorithms," Siam Review, 24(3), pp.275-287.

[2] Ernst, A.T., Jiang, H., Krishnamoorthy, M. and Sier, D., 2004. "Staff scheduling and rostering: A review of applications, methods and models," European journal of operational research, 153(1), pp.3-27.

[3] Van den Bergh, J., Belin, J., De Bruecker, P., Demeulemeester, E. and De Boeck, L., 2013. "Personnel scheduling: A literature review," European Journal of Operational Research, 226(3), pp.367-385.

[4] Loucks, J.S. and Jacobs, F.R., 1991. "Tour scheduling and task assignment of a heterogeneous work force: A heuristic approach," Decision Sciences, 22(4), pp.719-738.

[5] Vakharia, A.J., Selim, H.S. and Husted, R.R., 1992. Efficient scheduling of part-time employees. Omega, 20(2), pp.201-213.

[6] Alvarez-Valdes, R., Crespo, E. and Tamarit, J.M., 1999. "Labour scheduling at an airport refuelling installation," Journal of the Operational Research Society, 50(3), pp.211-218.

[7] Musliu, N., Grtner, J. and Slany, W., 2002. "Efficient generation of rotating workforce schedules," Discrete Applied Mathematics, 118(1), pp.85-98.

[8] Lau, H.C., 1996. "On the complexity of manpower shift scheduling," Computers & Operations Research, 23(1), pp.93-102.

[9] Musliu, N., Schaerf, A. and Slany, W., 2004. "Local search for shift design," European journal of operational research, 153(1), pp.51-64.

[10] Brusco, M.J. and Jacobs, L.W., 1993. "A simulated annealing approach to the solution of flexible labour scheduling problems," Journal of the Operational Research Society, 44(12), pp.1191-1200.

[11] Zolfaghari, S., El-Bouri, A., Namiranian, B. and Quan, V., 2007. "Heuristics for large scale labour scheduling problems in retail sector," INFOR: Information Systems and Operational Research, 45(3), pp.111-122.

[12] Di Gaspero, L., Grtner, J., Kortsarz, G., Musliu, N., Schaerf, A. and Slany, W., 2007. "The minimum shift design problem," Annals of operations research, 155(1), pp.79-105.

[13] Kortsarz, G. and Slany, W., 2001. "The minimum shift design problem and its relation to the minimum edge-cost flow problem," Unpublished manuscript.

[14] Kyngs, N., Goossens, D., Nurmi, K. and Kyngs, J., 2012. "Optimizing the unlimited shift generation problem," Applications of Evolutionary Computation, pp.508-518.

[15] Eveborn, P. and Rnnqvist, M., 2004. "Schedulera system for staff planning," Annals of Operations Research, 128(1), pp.21-45.

[16] Jaumard, B., Semet, F. and Vovor, T., 1998. "A generalized linear programming model for nurse scheduling," European journal of operational research, 107(1), pp.1-18.

[17] Mason, A.J. and Smith, M.C., 1998, July. "A nested column generator for solving rostering problems with integer programming," In International conference on optimisation: techniques and applications (pp. 827-834). Curtin University of Technology Perth, Australia.

[18] Burke, E.K., De Causmaecker, P., Berghe, G.V. and Van Landeghem, H., 2004. "The state of the art of nurse rostering," Journal of scheduling, 7(6), pp.441-499.

[19] Dowsland, K.A., 1998. "Nurse scheduling with tabu search and strategic oscillation," European journal of operational research, 106(2-3), pp.393-407.

[20] Aickelin, U. and Dowsland, K.A., 2004. "An indirect genetic algorithm for a nurse-scheduling problem," Computers & Operations Research, 31(5), pp.761-778.

[21] Bai, R., Burke, E.K., Kendall, G., Li, J. and McCollum, B., 2010. "A hybrid evolutionary approach to the nurse rostering problem," IEEE Transactions on Evolutionary Computation, 14(4), pp.580-590.

[22] sgeirsson, E.I., 2014. "Bridging the gap between self schedules and feasible schedules in staff scheduling," Annals of Operations Research, 218(1), pp.51-69.

[23] Cai, X. and Li, K.N., 2000. "A genetic algorithm for scheduling staff of mixed skills under multi-criteria," European Journal of Operational Research, 125(2), pp.359-369.

[24] Bartholdi III, J.J., 1981. "A guaranteed-accuracy round-off algorithm for cyclic scheduling and set covering," Operations Research, 29(3), pp.501-510.

[25] Pinedo, M. and Chao, X., 1999. Operations scheduling. McGraw Hill.

[26] Holland, J.H., 1992. "Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence," MIT press.

[27] Aickelin, U. and Dowsland, K., 2008. "Exploiting problem structure in a genetic algorithm approach to a nurse rostering problem," arXiv preprint arXiv:0802.2001.

[28] Srinivas, N. and Deb, K., 1994. "Muiltiobjective optimization using nondominated sorting in genetic algorithms," Evolutionary computation, 2(3), pp.221-248.

[29] Coit, D.W., Smith, A.E. and Tate, D.M., 1996. "Adaptive penalty methods for genetic optimization of constrained combinatorial problems," INFORMS Journal on Computing, 8(2), pp.173-182.

[30] Runarsson, T.P. and Yao, X., 2000. Stochastic ranking for constrained evolutionary optimization. IEEE Transactions on evolutionary computation, 4(3), pp.284-294.

[31] Coello, C.A.C., 2002. "Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art," Computer methods in applied mechanics and engineering, 191(11), pp.1245-1287.

[32] Ning, X. "A MILP model to the multi-objectives shift assignment problem," Technical report of automated scheduling, optimisation and planning (ASAP) Group, University of Nottingham.

[33] Lpsolve 5.5. Open source (Mixed-Integer) Linear Programming system. URL http://lpsolve.sourceforge.net/5.5/.

[34] Inc. Gurobi Optimization. Gurobi optimizer reference manual, 2017. URL http://www.gurobi.com.

[35] Davis, L., 1991. Handbook of genetic algorithms.

[36] Asensio-Cuesta, S., Diego-Mas, J.A., Cans-Dars, L. and Andrs-Romano, C., 2012. A genetic algorithm for the design of job rotation schedules considering ergonomic and competence criteria. The International Journal of Advanced Manufacturing Technology, 60(9-12), pp.1161-1174.

[37] Beddoe, G.R. and Petrovic, S., 2006. Selecting and weighting features using a genetic algorithm in a case-based reasoning approach to personnel rostering. European Journal of Operational Research, 175(2), pp.649-671.

[38] Elizondo, R., Parada, V., Pradenas, L. and Artigues, C., 2010. An evolutionary and constructive approach to a crew scheduling problem in underground passenger transport. Journal of Heuristics, 16(4), pp.575-591.

[39] Frey, L., Hanne, T. and Dornberger, R., 2009, May. Optimizing staff rosters for emergency shifts for doctors. In Evolutionary Computation, 2009. CEC'09. IEEE Congress on (pp. 2540-2546). IEEE.

[40] Hanne, T., Dornberger, R. and Frey, L., 2009, May. Multiobjective and preference-based decision support for rail crew rostering. In Evolutionary Computation, 2009. CEC'09. IEEE Congress on (pp. 990-996). IEEE.

[41] Lui, P. and Teodorovi, D., 2007. Metaheuristics approach to the aircrew rostering problem. Annals of Operations Research, 155(1), pp.311-338.

[42] Puente, J., Gmez, A., Fernndez, I. and Priore, P., 2009. Medical doctor rostering problem in a hospital emergency department by means of genetic algorithms. Computers & Industrial Engineering, 56(4), pp.1232-1242.

[43] Valls, V., Prez, . and Quintanilla, S., 2009. Skilled workforce scheduling in service centres. European Journal of Operational Research, 193(3), pp.791-804.

[44] Wu, M.C. and Sun, S.H., 2006. A project scheduling and staff assignment model considering learning effect. The International Journal of Advanced Manufacturing Technology, 28(11-12), pp.1190-1195.

[45] Manuel Lpez-Ibez, Jrmie Dubois-Lacoste, Leslie Prez Cceres, Thomas Sttzle, and Mauro Birattari. The irace package: Iterated Racing for Automatic Algorithm Configuration. Operations Research Perspectives, 3:43-58, 2016.