A Hybrid Metaheuristic Approach to a Real World Employee Scheduling Problem

Kenneth N. Reid
Ken@kenreid.co.uk
University of Stirling
Stirling, UK

Jingpeng Li
jli@cs.stir.ac.uk
University of Stirling
Stirling, UK

Alexander Brownlee
sbr@cs.stir.ac.uk
University of Stirling
Stirling, UK

Mathias Kern
mathias.kern@bt.com
BT Applied Research
Ipswich, UK

Nadarajen Veerapen
nadarajen.veerapen@univ-lille.fr
Univ. Lille, CNRS, Centrale Lille,
UMR 9189 – CRIStAL
Lille, France

Jerry Swan
dr.jerry.swan@gmail.com
University of Stirling
Stirling, UK

Gilbert Owusu
gilbert.owusu@bt.com
BT Applied Research
Ipswich, UK

ABSTRACT

Employee scheduling problems are of critical importance to large businesses. These problems are hard to solve due to large numbers of conflicting constraints. While many approaches address a subset of these constraints, there is no single approach for simultaneously addressing all of them. We hybridise 'Evolutionary Ruin & Stochastic Recreate' and 'Variable Neighbourhood Search' metaheuristics to solve a real world instance of the employee scheduling problem to near optimality. We compare this with Simulated Annealing, exploring the algorithm configuration space using the `irace` software package to ensure fair comparison. The hybrid algorithm generates schedules that reduce unmet demand by over 28% compared to the baseline. All data used, where possible, is either directly from the real world engineer scheduling operation of around 25,000 employees, or synthesised from a related distribution where data is unavailable.

CCS CONCEPTS

• **Computing methodologies** → **Randomized search**; • **Applied computing** → **Industry and manufacturing**;

KEYWORDS

Evolutionary Ruin and Stochastic Recreate, Metaheuristics, Employee Scheduling, Variable Neighbourhood Search

1 INTRODUCTION

In this paper we present a solution to a real world employee scheduling problem, using a hybridization of Evolutionary Ruin & Stochastic Recreate (ER&SR) with Variable Neighbourhood Search (VNS) and an evolutionary Monte-Carlo acceptance criterion (EMCAC).

The problem specification and associated data were provided by a FSO (Field Service Operations company) and specific problem details are intentionally omitted. This collaboration with the FSO ensured the precise design of constraints and fitness criteria. Since this paper deals with a real world problem, the algorithm's effectiveness is empirically grounded.

Employee scheduling is a problem where employees are assigned to one or more shifts in order to meet a number of objectives (e.g. improving employee efficiency in their roles and meeting demand). Employee scheduling is a top concern for management in improving organisational efficiency [8]: failure to optimize shift allocation for employees can mean a loss in profits, customer satisfaction and failure to meet demand.

ER&SR and VNS have been successful when used individually to produce satisfiable schedules in the domain of personalised scheduling [22, 23]. Our proposed approach in this paper compared three combinations of VNS and ER&SR, and uses EMCAC to control the end criterion. The three approaches are as follows: ER&SR followed sequentially by VNS (i.e. each algorithm runs only once), an ER&SR and VNS hybrid (VNS within ER&SR) and a third approach which combines an internal VNS and a sequential VNS with ER&SR. This is detailed further in Section 4.

The `irace` package [18] was used to automatically configure the parameters for the various ER&SR and VNS settings. This is to ensure comparisons were fair with the same time budgets for each configuration, excluding the handwritten baseline.

The contributions of this paper are as follows:

(1) Propose the new hybridised algorithm of ER&SR and VNS.
(2) Analysis of this algorithm's application to a real world employee scheduling problem.
(3) A systematic parameter search by `irace` to find a suitable configuration.
(4) A comparison of the hybrid algorithm with four other configurations and the improvement relative to the baseline.

The problem domain is similar to that considered in a previous work [23], in which both the shift scheduling and employee rostering problems with personalised schedules are tackled. A personalised schedule is unlike a Roster Pattern (RP) based approach, in that shift start times, end times, shift lengths are variable day to day. In this paper we utilise a RP based approach where a set of shift patterns are input parameters and cannot be modified, instead the selected pattern per employee, and start week, are modified.

The base problem requires the assignment of a skill for each employee, for each shift. The additional constraints are expressed in the form of employee RPs, which requires the additional week-wise assignment of a RP an employee. The search space cannot be exhaustively explored in polynomial time and similar employee scheduling problems have been considered NP-Hard [14]. Metaheuristic solutions were used to solve this problem to acceptable quality in reasonable business time frames, as defined by the FSO. We propose a hybrid solution in anticipation of jointly benefiting from the exploratory nature of ER&SR and the exploitative local search of VNS.

Employee rostering is often used interchangeably with employee scheduling and workforce rostering. In this paper we use the term 'employee rostering' to mean the allocation of an employee to a shift or set of shifts which are preset and immutable. In this paper we also use the term 'employee' and 'engineer' interchangeably as the real world problem pertains to engineers.

This paper is structured as follows: Section 1 completes with a discussion on related works in subsection 2. In Section 3 the problem description is provided as are the benefits of this approach. Section 4 details the solution implementation, using the theoretical framework provided by Li et al [17] as a basis, and the unique elements of our implementation are described. The results of this approach are analysed in Section 5 and Section 6 presents conclusions.

2 RELATED WORK

In the field of employee rostering there is much motivation for obtaining high quality employee rosters. Managing the shifts of a large number of employees is a complex task — optimisation of this process is a necessity for efficient use of resources. The field of employee rostering has been extensively surveyed: one of the first surveys of the state of the art was produced by Baker [1] in 1976. In 2004 Ernst et al produced a review of applications, methods and models [10], which not only successfully highlighted some of the most influential research in the field, but also correctly predicted future trends. Such trends include employee scheduling in airlines, which has since been further examined by several reports [2] [4], and the need for more robust frameworks to tackle the complexity of these problems.

The extensive 2013 review 'Personnel Scheduling: A Literature Review' by Van Den Bergh et al [3], gives a thorough exploration of the field, in which papers are categorised by criteria such as personnel characteristics, decision types, shift flexibility, coverage constraints (both hard and soft), whether or not under- and overstaffing is allowed, by skills, and more. There have been several other notable domain specific literature reviews, such as home care routing and scheduling [11] and physician scheduling [9].

The term 'metaheuristic' was first coined by Glover [13] in the celebrated paper introducing Tabu search. Metaheuristics are high-level procedures for exploring complex solution spaces, designed to solve the problem of entrapment within local optima [12]. Metaheuristics are 'higher level' heuristics in that they define generic solution mechanisms: for example, a metaheuristic framework for Simulated Annealing can be customised for different problem domains (e.g. Travelling Salesperson, Vehicle Routing, Scheduling etc.) via the provision of domain-specific heuristics for solution neighborhood and solution quality. Popular metaheuristics include Simulated Annealing[16], VNS [19], tabu search [13], genetic algorithms, memetic algorithms [7] and many others.

Scheduling vast numbers of employees while taking demand, location, skill sets, preferences and contract types into consideration is a complex task which cannot be completed manually for a real-world workforce. It is often impossible to exhaustively search for an optimal solution in a timely fashion [5]. As such other approaches have been developed by operational engineers, computer scientists and employee managers to generate optimal or close-to-optimal solutions, e.g. using evolutionary computation approaches [20]. Improving labour costs by only a few percent could prove very beneficial, since this is often the main expenditure for companies [3].

Rather than the expensive (and often impossible) alternative of analysing every potential roster, an evolutionary algorithm can sample the solution space for a result which is often at least close-to-optimal, within a reasonable time. VNS is a methodology which is also commonly used in optimisation problems. In a nurse rostering problem example [21], VNS is hybridized with integer programming. In this approach a greedy heuristic is employed to tackle the initial problem and create a base solution, which is successfully improved upon iteratively with VNS deep-embedded with integer programming. Bruecker et al in a 2015 state-of-the-art review [6] on workforce planning incorporating skills concluded that more real world application is essential to advance the field. The authors also found that mixed integer linear programming appears to be the most popular mathematical programming approach, while custom approaches are the most popular in heuristic methodologies.

3 PROBLEM DESCRIPTION

In this section we describe the problem as defined by collaboration with the FSO. We then state the Hard Constraints (HCs) and Soft Constraints (SCs), giving context for a quality measure on feasible solutions.

Similarly complex employee scheduling problems in the literature are NP-Hard [14], so the goal of this research is to find a solution meeting all HCs and as many SCs as possible within a time span deemed acceptable by the FSO. This time span is within 2 minutes on a medium specification workbook issued to employees within the FSO. For testing purposes run time is greatly increased, as described in Section 5.

The problem and data are both real world and provided by an FSO company. The information in the paper has been anonymised and generalised for security and data protection reasons.

The problem we tackle in this paper schedules up to 200 employees, varying depending on input data region, from a pool of around 25,000 employees across the UK. There are therefore hundreds of possible sets of input data from various localities. There are a variety of local differences which change frequently due to variation in demand forecasting or problem region. Differences can include different temporal patterns in demand, different skills required, different skills available from engineers and a different number of engineers available. This problem has the following unique combination of traits as defined in collaboration with the FSO:

(1) Due to the real-world data being provided by a UK based FSO, UK Employment Law must be adhered to. This includes

maximum working hours allowed per week, number of rest hours in between shifts, etc.

(2) The employees described in the data have a variety of skills. Demand is provided for the problem in skill-based manner. Demand may fluctuate greatly day to day, week to week. The objective of meeting demand includes not simply providing manpower to cover number of employees required per shift, but skills required per shift, and per day, while still meeting HCs.

The term "shift" refers to a time period within some 24 hour period. A shift can have a single employee allocated to it. Shifts have the attributes of start time, end time, length, demand covered (in terms of an integer array with potential values of 0 or 1 per hour) and skill covered.

A solution is deemed feasible if the associated schedule does not violate any legal, contractual or ethical guideline defined by the FSO, each of which is represented by a HC. Such guidelines prevent, for example, working illegal number of hours per week or working shifts that violate an employee's contract. As this problem works from RPs provided by the FSO, most HCs that would normally be considered in employee scheduling are unnecessary, as every RP meets all legal and contractual obligations. HCs therefore cannot be violated by nature of the encoding of the problem: the algorithm will only generate and test solutions which meet all HCs. Thus the concern of this paper is only to minimise the cost of SCs, or inversely improve the quality (fitness) of schedules.

- HC1: Employees must have exactly one RP.
- HC2: An optionally specified maximum number of RP changes can occur.
- HC3: An optionally specified maximum number of starting week changes can occur.
- HC4: If RP preferences are provided, only RPs preferred by each employee can be explored.

HC1 is hardcoded, an employee cannot be given more than or less than one RP outside of a single iteration of ER&SR or VNS (meaning that employees do have RPs removed, but they are then given another RP, or potentially the same one again).

HC2 and HC3 can optionally have the maximum number of changes set to no maximum, zero (meaning no changes of this type allowed) or a specific number. An example could be maximum of 10 RP changes allowed. When 10 have occurred, only those modified RPs can be changed. If one of these is returned to the original pattern, then the current number of changes is reduced to 9, and another employee's pattern could potentially be changed. In this paper HC2 and HC3 are set to infinite, meaning they are always satisfied.

HC4 can be set to on or off. If set to off, any employee could potentially be given any RP. This is generally not realistic in many regions as certain RPs are created for special circumstances, specific contractual requirements, etc. It would also mean employees could be given the incorrect number of working hours per week. However, if the available RPs are all verified to be interchangeable, then this is not an issue.

For most regions, employees have a number of preferred RPs, and HC4 would be toggled on. The preferred patterns will always meet contractual obligations. These patterns are agreed upon by both employee and regional managers. Every employee has at least one preferred RP, which is their current pattern. There is no maximum number of preferred RPs, other than the number available in the pool, which is dependent on how many have been created by managers in the past. In the region used for this paper there are around 400 potential RPs employees could have in their preferred patterns list.

There are no hard constraints regarding starting week optimisation, so any employee can still have rotas modified, as well as skills worked every shift even if they cannot change RP directly. For example, if an employee has a rotating rest day off (Monday on week 1, Tuesday on week 2, ..., Saturday on week 6) then changing the starting week can drastically affect potential skills available on week days across a number of months.

A solution's quality is judged with SCs. SCs consider elements that are important for the quality of life for engineers, for meeting demand and for meeting customer satisfaction. These are important, but do not break any laws or contractual requirements if not met. The following SCs are considered to judge the quality of a solution:

- SC1: Shifts should cover the maximum demand requirement they possibly can.
- SC2: Employees should be assigned to shifts that require a skill listed in their preferences.
- SC3: Allocations of employees to shifts should favour employee skill preferences in descending order.
- SC4: Variation in the percentage of demand met per hour, day and skill should be minimised.

To be clear: all engineers have the skills required for all work, however employees have a list of "preferred" skills. The reasons for the skills being preferred are varied, from personal enjoyment, to proficiency or managerial preference. As such, SC2 measures whether employees are working a skill listend, SC3 judges quality based on level of preference.

Combined weighted constraint violations are used to judge solution fitness. Specific weightings are provided by the FSO; in this paper all weightings are equal (25% per SC). In real world usage the weightings are dependent on locale specific priorities, equal weighting was judged to be acceptable for testing within this paper. As an example of why this is useful, if demand is not being met to a satisfiable level, SC1 could be set to 100%, and the others set to 0%.

4 IMPLEMENTATION

4.1 Evolutionary Ruin & Stochastic Recreate

ER&SR [17] is a variant of the original Ruin & Recreate method [24] by the addition of evolutionary features, selection and mutation, as the ruining strategies. In this section we describe the differences from previous work on ER&SR. ER&SR was used to solve not only an employee rostering but also a shift scheduling problem, without VNS in previous works [23], however in this instance ER&SR is used in hybrid with VNS to solve a real world employee rostering problem, not a shift scheduling problem.

Our implementation was designed to be modular, with a VNS component which can be called at any stage of the algorithm. This allows us to consider a variety of approaches and how to best exploit the local search advantages from VNS in conjunction with the more exploratory ER&SR. As such we implemented the following hybridised algorithms:

Listing 1: ER&SR Pseudocode

```
1   ERSR(vnsInternalIterations,
2               vnsSequentialIterations) {
3       a = originalSolutionFromFSO
4       //Exponential Monte-Carlo Acceptance
5       while(currentTemperature > delta){
6           b = a.copy()
7           fitness = decomposition(softConstraintWeightings)
8           //Evolutionary Ruin
9           a = selection(a, selectionRate)
10          a = mutation(a, mutationRate)
11          //Stochastic Recreate
12          a = rebuild(a)
13          //Solution Acceptance
14          p = getAcceptanceProbability()
15          if(p < random.nextInt()){
16              a = b.copy()
17              if(vnsInternalIterations > 0){
18                  beginVNS()
19              }
20          }
21          currentTemperature *= coolingRate
22      }
23      if(vnsSequentialIterations > 0)
24          beginVNS()
25  }
```

(i) ER&SR with VNS running sequentially after ER&SR stopping criterion (referred to as 'ER&SR with VNS Sequential')
(ii) ER&SR with VNS running after accepting a modified schedule (referred to as 'ER&SR with VNS Internal')
(iii) ER&SR with both the above (referred to as 'ER&SR with both VNS internal and sequential')

Pseudocode of ER&SR with VNS can be seen in Listing 1. Lines 19 and 25 reference VNS, of which is described more fully in subsection 4.2, with VNS pseudocode available in Listing 2.

Our previous paper [23] worked with a personalised scheduling problem as opposed to RPs, we introduced a number of changes. The same phases exist and perform the same functions with modifications, and the VNS hybridisation has since been implemented.

(1) The initial set up phase initialises the software and reads the current hand-written solution in place by the FSO.
(2) The EMCAC control mechanism is initialised with the parameterised inputs, controlling the number of times ER&SR loops (but not the sequential VNS element).
(3) Phase 1 - solution decomposition no longer considers HCs. This is due to RPs which ensure contractual and legal requirements cannot be breached. More precisely, the HCs considered in this paper can be temporarily breached during the evolutionary ruin phase, but are all then restored during stochastic recreate. SCs are considered in the solution decomposition, and the fitness evaluations can be completed by considering only the potential changes from the current state (referred to by [17] as componential fitness). This is necessary as all four SCs consider the percentage of all individual shifts meeting critera. The solution decomposition serves the purpose of analysing fitness during ER&SR run-time as well as after VNS swaps have occurred to decide whether to keep or revert changes.
(4) Phase 2 - evolutionary ruin is mostly unchanged. Instead of ruining individual shifts, both the selection and mutation elements now ruin employee RPs, by giving a null allocation. Selection will destroy elements with the following probability:

$$P(s) = r * x_1 ,$$

where $P(s)$ is the probability of acceptance, r is a random number between 0 and 1, and x_1 is a parameter to limit the stochasticity of selection. If P is higher than the fitness of the employee allocated to their current RP, then the employee has their RP removed for a new one. After selection is complete, mutation considers all non-affected employees, and randomly removes employees from RP. This adds an additional level of stochasticity to the search. Fitness is not considered during mutation. Mutation can be overly destructive and hinder exploitation, so careful parameterisation is important. The chance of ruination is calculated as follows:

$$P(b) = r * x_2 ,$$

where $P(b)$ is the probability, r is a random number between 0 and 1, x_2 is the parameter used to increase or decrease probability. All of this phase is likely to break all hard constraints before the schedule is reconstructed and made feasible in the next phase.
(5) Phase 3 - stochastic recreate no longer finds random shifts to allocate employees to, but instead allocates employees to random RPs and starting weeks. Employees to be allocated a new shift pattern are chosen randomly. This is so the search space can be explored if a maximum bound is set via HC2 or HC3. These HCs are considered with every RP allocation to ensure the maximum bound is not breached. If HC4 is toggled on, only RPs the employee has selected may be considered for allocation. Otherwise, any RP can be allocated.
(6) Phase 4 - solution acceptance will probabilistically accept the changes. This is unchanged from the above mentioned previous work, and still uses the EMCAC. The acceptance probability is calculated as follows:

$$P = \exp((o - n)/t) > r ,$$

where P is the probability of accepting a worsened state, o is the original solution energy cost (modelled as inverse fitness), n is the energy cost of the new solution, t is current temperature and r is a uniformly generated random number between 0 and 1.

4.2 Variable Neighbourhood Search

After the ER&SR algorithm has accepted a new solution, VNS is called (if parameters indicate to do so). The VNS method chosen is basic VNS [12]. While more complex varieties of VNS are available (such as other hybrid solutions [21], or with adaptive memory improvements [25]), ER&SR already provides a feasible close-to-optimal solution. Therefore VNS is used to search nearby neighbourhoods for minor improvements rather than building from the baseline.

Assuming iterations left is above zero, VNS first finds the fitness of the current schedule. Each employee defines a neighbourhood. A random employee is selected, and so is their RP. If HC4 is enabled, only RPs the employee has a preference for are considered for the next step, otherwise all RPs are considered. A random RP from the selected list is chosen. Checking if no HCs are failed by a potential swap, then the swap goes ahead. Then fitness of this new schedule

Listing 2: VNS Pseudocode

```
VNS(k_max, t_max){
    t = 0
    while(t < t_max){
        //fitness is parameterised weighted SCs
        f = getFitness()
        //neighbourhood is allowed RPs for a
        a = randomEmployee()
        //RP = roster pattern
        s = getRP(a)
        k = 0
        while(k < k_max){
            s' = null
            //Only consider allowed RPs for a.
            if(HC4isTurnedOn())
                s' = RPFromPreferences(a)
            //consider all RPs.
            else
                s' = randomRP()
            if(swapWouldBreakHC2orHC3())
                continue
            giveEmployeeRP(a, s')
            f' = getFitness()
            //if fitness is reduced, undo.
            if(f' < f)
                giveEmployeeRP(a, s)
            k++
        }
        t++
    }
}
```

is determined. If the new fitness is lower than the original, then the schedule is reverted, otherwise it is kept. See Listing 2 for pseudocode of the VNS algorithm.

4.3 Order of Events

After the internal VNS has completed, ER&SR repeats until the completion criterion is met. VNS can then run after ER&SR has completed, if parameterised. This would allow the best solution found by ER&SR to be exploited further by VNS. In this paper we apply VNS either a) during ER&SR (shortly after a solution has been accepted) for a number of parameterised iterations, b) after the ER&SR iterations have all completed, or c) both. This means this algorithm can be run with ER&SR only, ER&SR with VNS Internal, ER&SR with VNS Sequential, and ER&SR with both.

Fig 1 provides a visual overview of the hybrid algorithm. In the diagram, the internal VNS (to the left) is shorthand for the same process to the right of the diagram. Both the internal VNS and the sequential VNS are only active if parameterised to run.

5 RESULTS

In this section we discuss results from tests run on an Intel i5 2.49GHz CPU. The configurations tested were:

(A) ER&SR with both VNS Internal and VNS Sequential.
(B) ER&SR with VNS Internal.
(C) ER&SR with VNS Sequential.
(D) ER&SR only.
(E) VNS only.
(F) Simulated Annealing only.
(G) Baseline

The baseline is also shown in the results for comparison (the baseline is the fitness computed from the manually generated solution in place at the FSO). The baseline solution is always feasible.

The effectiveness of metaheuristics are largely dependent on well-tuned parameters [5]. This is particularly important in this instance as there are many parameters and ranges to select from. Thus the parameters for our testing were chosen by the `irace` parameter configuration tool [18], such automated parameter tuning being preferable to onerous manual configuration [15]. During `irace` testing, the testing of elite configurations was enabled, meaning that only the best configurations found during the run of `irace`, the final best, will be used in the testing phase. The first elimination test value was set to the default of five. The default statistical test configuration for `irace` was used (F-test).

Each configuration had 30 iterations within `irace`. The parameters tuned by `irace` are listed in Table 1; temperature, cooling rate, selection rate, mutation phase 1 rate, mutation phase 2 rate, VNS internal iterations, and VNS sequential iterations. Not all of the parameters were tuned for every algorithm configuration: e.g. cooling rate does not apply to 'VNS only' or the baseline; vnsintr only applies when VNS Internal is enabled. Note that Table 1 shows the parameters for `irace` for a single test (specifically ER&SR with VNS internally), as an example.

These dependencies are reflected in Table 2, the tuned configurations found by `irace`. Here, 'N/A' is used when a variable is not tuned. The cooling rate was limited to a maximum of 0.5 due to previous observations during testing finding that too few iterations caused by a higher cooling rate caused poorer results. Similarly the VNS internal number of iterations was capped at 100, as higher values caused a dramatic increase in run-time.

After the parameters were tuned with `irace`, the algorithms were then each run, with tuned parameters, thirty times. The mean values for these tests can be seen in Table 3. A box-plot showing the range of fitness values can be viewed in Fig. 2. The results shown in this paper reference a single region and around 180 employees being scheduled. HC2 and HC3 have no maximum bound, allowing any number of changes — which both increases the complexity of the problem but also increases the chance of finding better soft constraint compliant allocations. The demand forecast data used for testing is provided by the FSO is calculated by an internal research team for the region the employees tested work.

The SA metaheuristic, given by Config F, has the most variability in terms of fitness provided by the soft constraints. Across all of the configurations that use ER&SR, interestingly there is no solution generated which is worse than the baseline, but that is likely due to the high number of iterations present in the run configuration parameters provided by `irace` (both directly by VNS iterations and indirectly via temperature & cooling rate values).

A non-parametric ANOVA test (Kruskal-Wallis) was applied to the results and returned a p-value of $p < 2.2e^{-16}$. The datasets were confirmed visually as normally distributed in histogram plots as well as in Q-Q plots, which is a plot of the quantiles of the two distributions. The Q-Q plot of Configuration A can be viewed at Figure 3. We can reject the null hypothesis that the results for each configuration are drawn from the same distribution.

There is clear indication that ER&SR provides better schedules, and more so when using the VNS module at any tested stage, than

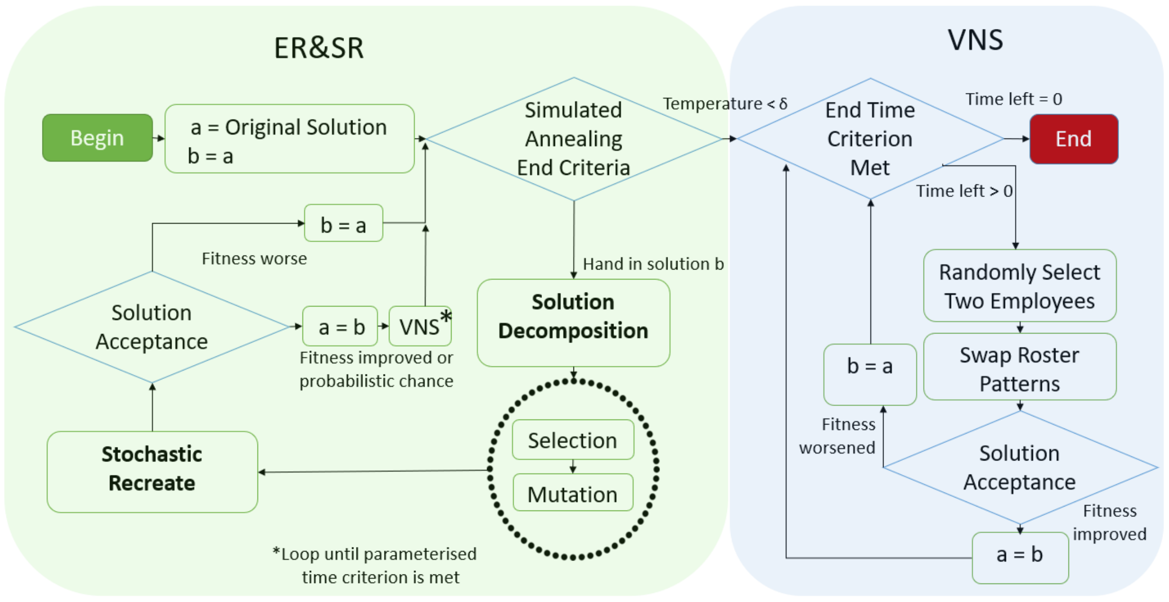

Figure 1: Diagrammatic Overview of ER&SR & VNS Hybrid

Parameter	Reference	Type	Range	Additional Parameters
temp	−a	r	(1,1000000)	
cr	−b	r	(0.0001,0.5)	
sr	−c	r	(0.0001,1.0)	
mp1r	−d	r	(0.0001,1.0)	
mp2r	−e	r	(0.0001,1.0)	
vnsint	−f	c	(“on")	
vnsintr	−g	r	(1.0,100.0)	| vnsint %in% c(“on")
vnsseq	−h	c	(“off")	
vnsseqr	−i	r	(1.0,2000.0)	| vnsseq %in% c(“on")

Table 1: Parameters To Be Tuned by irace

Config	Temp	Cooling Rate	Selection Rate	Mutation P1 Rate	Mutation P2 Rate	VNS Internal	VNS Sequential	Simulated Annealing
A	**116621.6**	**0.045**	**0.434**	**0.041**	**0.353**	**35.951**	**743.145**	**N/A**
B	33904.68	0.221	0.707	0.045	0.597	81.053	N/A	N/A
C	711429.8	0.024	0.301	0.791	0.486	N/A	103	N/A
D	706984.6	0.007	0.881	0.721	0.696	N/A	N/A	N/A
E	N/A	N/A	N/A	N/A	N/A	N/A	1963	N/A
F	9189272	0.004	N/A	N/A	N/A	N/A	N/A	1
G	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

Table 2: Tuned Parameters

the schedules created by SA, or by VNS alone. The results show that the method this paper provides is an improved alternative to current RPs / starting weeks employees are working on the baseline solution for this dataset and demand forecast. The schedules are improved in terms of average soft constraint fitness, which is how the quality of a solution is judged. View Table 4 for comparison of demand hours met, which shows unmet demand is reduced by over 28% from Config A compared to the baseline of Config G.

The results also show that this algorithm can provide a small but relatively quick improvement upon standard ER&SR, where

Config	ER&SR	VNS Internal	VNS Sequential	Simulated Annealing	Average Fitness
A	**1**	**1**	**1**	0	**0.55993**
B	1	1	0	0	0.54985
C	1	0	1	0	0.54227
D	1	0	0	0	0.51884
E	0	0	1	0	0.49834
F	0	0	0	1	0.44840
G	0	0	0	0	0.42987

Table 3: Average Fitness Per Run Type

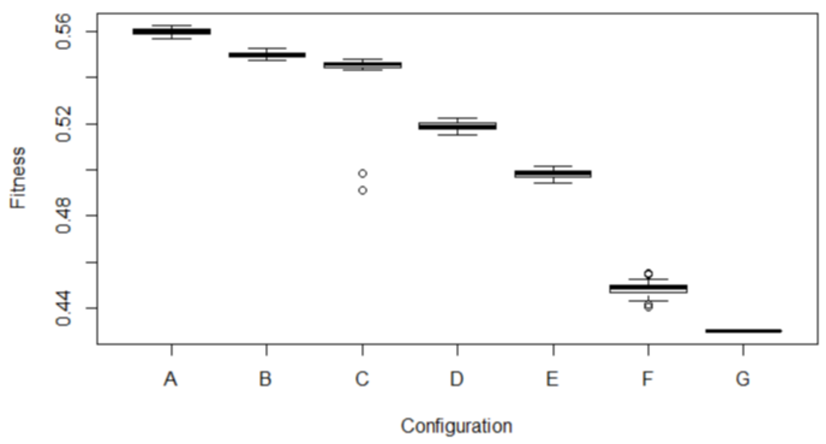

Figure 2: Box-Plot of Average Fitness (A-F) and Baseline Fitness (G)

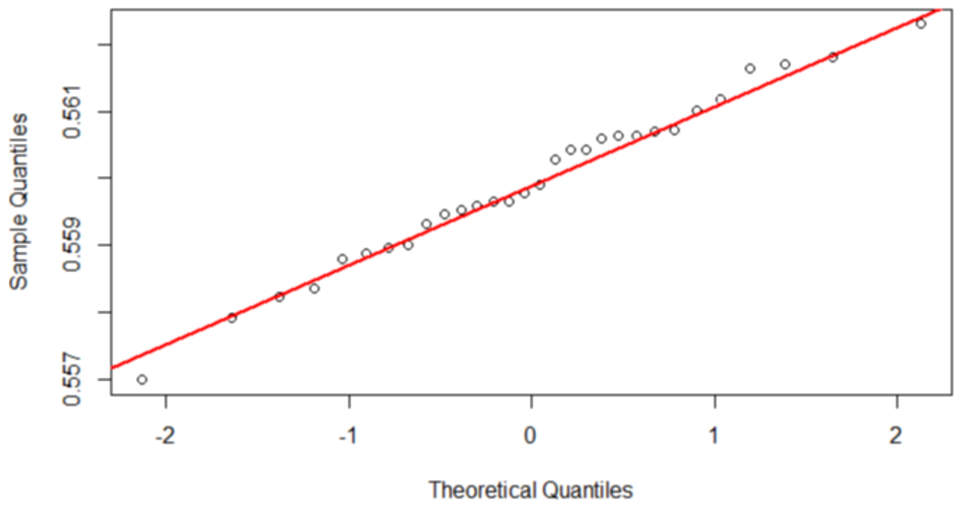

Figure 3: Q-Q Plot of Config. A

using both internal and sequential VNS have the highest rate of improvement, on this dataset. The results show that demand met is improved on average, based on an estimated demand forecast from the FSO.

6 CONCLUSIONS

This paper presents a hybrid algorithm combining Evolutionary Ruin & Stochastic Recreate (ER&SR) with Variable Neighbourhood Search (VNS) and compares it to a Simulated Annealing (SA) approach and a first-improvement VNS approach. We apply this to

Config	Original Demand	Demand Hours Met	Demand Hours Not Met
A	**80305**	**55426**	**24879**
B	80305	55198	25107
C	80305	55180	25125
D	80305	54699	25606
E	80305	52980	27325
F	80305	51512	28793
G	80305	45373	34932

Table 4: Demand Met Per Config

a real world problem provided by a Field Service Operations company solving the employee rostering problem. The novelty of this approach lies in the improvement of the solution compared to standalone ER&SR. A further novelty is application of hyper-parameter tuning to ER&SR (via the `irace` algorithm configuration tool). This solution also provides an improved schedule for a real world application when compared to the approach currently in place.

There is also scope for further research on ER&SR, in particular when hybridised with industry solvers such as CPLEX, which would include a mathematical formulation of the problem. There is also potential in replacing the stochastic rebuild section of ER&SR with a different mechanism, for example with an exact method with lowered bounds, other metaheuristics or matheuristics. While in this paper the number of VNS iterations are auto configured by `irace`, an interesting future development may be a reverse EMCAC to control the depth of the search of the VNS, so that more exploitative search can take place towards the end of run-time. This may be particularly useful as it was found during experimentation that the ruin method is very destructive early in run-time, negating any benefits of VNS. Additionally, future work should provide random seed trial search dynamics across methods tested to improve replicability of these stochastic algorithms.

7 DATA ACCESS STATEMENT

The raw employee data nor roster pattern intellectual property cannot be shared due to GDPR and intellectual property laws.

8 ACKNOWLEDGEMENTS

The authors of this paper extend gratitude to those involved and the following funders: British Telecommunications Plc and the UK's EPSRC DAASE project (grant no. EP/J017515/1). J. Swan acknowledges the support of the EU H2020 SAFIRE project (Ref. 723634). Additional thanks to Dr Paul McMenemy for additional support.

REFERENCES

[1] Baker, K.R.: Workforce allocation in cyclical scheduling problems: A survey. Journal of the Operational Research Society **27**(1), 155–167 (1976)
[2] Benlic, U., Burke, E.K., Woodward, J.R.: Breakout local search for the multi-objective gate allocation problem. Computers & Operations Research **78**, 80–93 (2017)
[3] Van den Bergh, J., Beliën, J., De Bruecker, P., Demeulemeester, E., De Boeck, L.: Personnel scheduling: A literature review. European Journal of Operational Research **226**(3), 367–385 (2013)
[4] Burke, E.K., De Causmaecker, P., De Maere, G., Mulder, J., Paelinck, M., Berghe, G.V.: A multi-objective approach for robust airline scheduling. Computers & Operations Research **37**(5), 822–832 (2010)
[5] Burke, E.K., Kendall, G., et al.: Search methodologies. Springer (2005)
[6] De Bruecker, P., Van den Bergh, J., Beliën, J., Demeulemeester, E.: Workforce planning incorporating skills: State of the art. European Journal of Operational Research **243**(1), 1–16 (2015)
[7] El-Yaakoubi, A., El-Fallahi, A., Cherkaoui, M., Hamzaoui, M.R.: Tabu search and memetic algorithms for a real scheduling and routing problem. Logistics Research **10**(1), 7 (2017)
[8] Enz, C.: Key issues of concern in the lodging industry: what worries managers. Cornell Hospitality Report **9**(4), 1–17 (2009)
[9] Erhard, M., Schoenfelder, J., Fügener, A., Brunner, J.O.: State of the art in physician scheduling. European Journal of Operational Research (2017)
[10] Ernst, A.T., Jiang, H., Krishnamoorthy, M., Sier, D.: Staff scheduling and rostering: A review of applications, methods and models. European journal of operational research **153**(1), 3–27 (2004)
[11] Fikar, C., Hirsch, P.: Home health care routing and scheduling: A review. Computers & Operations Research **77**, 86–95 (2017)
[12] Gendreau, M., Potvin, J.Y.: Handbook of metaheuristics, vol. 2. Springer (2010)
[13] Glover, F.: Future paths for integer programming and links to artificial intelligence. Computers & operations research **13**(5), 533–549 (1986)
[14] Heimerl, C., Kolisch, R.: Scheduling and staffing multiple projects with a multi-skilled workforce. OR spectrum **32**(2), 343–368 (2010)
[15] Hoos, H.H.: Programming by optimization. Communications of the ACM **55**(2), 70–80 (2012)
[16] Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by simulated annealing. science **220**(4598), 671–680 (1983)
[17] Li, J., Bai, R., Shen, Y., Qu, R.: Search with evolutionary ruin and stochastic rebuild: A theoretic framework and a case study on exam timetabling. European Journal of Operational Research **242**(3), 798–806 (2015)
[18] López-Ibáñez, M., Dubois-Lacoste, J., Cáceres, L.P., Birattari, M., Stützle, T.: The irace package: Iterated racing for automatic algorithm configuration. Operations Research Perspectives **3**, 43–58 (2016)
[19] Mladenović, N., Hansen, P.: Variable neighborhood search. Computers & operations research **24**(11), 1097–1100 (1997)
[20] Pinedo, M., Zacharias, C., Zhu, N.: Scheduling in the service industries: An overview. Journal of Systems Science and Systems Engineering **24**(1), 1–48 (2015)
[21] Rahimian, E., Akartunalı, K., Levine, J.: A hybrid integer programming and variable neighbourhood search algorithm to solve nurse rostering problems. European Journal of Operational Research **258**(2), 411–423 (2017)
[22] Reid, K.N., Li, J., Swan, J., McCormick, A., Owusu, G.: Variable neighbourhood search: A case study for a highly-constrained workforce scheduling problem. In: Computational Intelligence (SSCI), 2016 IEEE Symposium Series on. pp. 1–6. IEEE (2016)
[23] Reid, K.N., Li, J., Veerapen, N., Swan, J., McCormick, A., Kern, M., Owusu, G.: Shift scheduling and employee rostering: An evolutionary ruin & recreate solution. In: 10th Computer Science and Electronic Engineering Conference (CEEC). IEEE (2018 (to appear))
[24] Schrimpf, G., Schneider, J., Stamm-Wilbrandt, H., Dueck, G.: Record breaking optimization results using the ruin and recreate principle. Journal of Computational Physics **159**(2), 139–171 (2000)
[25] Simeonova, L., Wassan, N., Salhi, S., Nagy, G.: The heterogeneous fleet vehicle routing problem with light loads and overtime: Formulation and population variable neighbourhood search with adaptive memory. Expert Systems with Applications (2018)