Sequential Meta-Heuristic Approach for Solving Large-Scale Ready-Mixed Concrete–Dispatching Problems

Mojtaba Maghrebi, M.ASCE[1]; S. Travis Waller, M.ASCE[2]; and Claude Sammut[3]

Abstract: Finding a practical solution for the allocation of resources in ready-mixed concrete (RMC) is a challenging issue. In the literature, heuristic methods have been mostly used for solving the RMC problem. The introduced methods are intended to find a solution in one stage but the amount of infeasible allocations in their initial solutions is their main challenge, as these infeasible solutions need postprocessing efforts. This paper introduces a sequential heuristic method that can solve RMC problems in two separate stages without any need for postprocessing. It was found that the depot-allocation problem is more complicated than truck allocation and the combination of these two subproblems threatens the efficiency of the solution. Another contribution of this paper is proposing a new formulation for minimizing the number of trucks. A genetic algorithm (GA) has been selected for implementing the proposed idea and for evaluating the large-scale data-set model. The data set covers an active RMC for a period of 1 month. The comprehensive tests show that sequential GA is more robust than traditional GA when it converges 10 times faster with achieved solution at 30% less cost. **DOI: 10.1061/(ASCE)CP.1943-5487.0000453.**

Introduction

In a ready-mixed concrete (RMC), typically fresh concrete is automatically mixed based on orders and is then loaded into trucks. A large-scale RMC has several batch plants for mixing concrete and several trucks with different sizes for supplying the fresh concrete to the customers. In a RMC-dispatching room, decisions are made to supply concrete to each customer from an available source and with a specific truck. The main objective of dispatching is to find an effective way of allocating resources at least cost. Theoretically, this resource-allocation problem can be modeled mathematically. However, acquiring the optimum solution in such a complex and large-scale problem is computationally intractable and characterized as a classic nondeterministic polynomial-time hard (NP hard) problem. This means that no existing solution can solve these problems in polynomial time. To overcome this problem, heuristic methods have been widely used in the literature, which will be comprehensively discussed in the following section. The quality of the introduced heuristic methods cannot be assessed due to the lack of the optimum solution. However, by comparing the introduced heuristic methods, more effective techniques can be found based on their performances. Unlike most of the similar introduced techniques in the literature, it is expected that the proposed sequential technique solves the RMC-dispatching problem accurately and fast as well as without any need for postprocessing efforts.

[1]Ph.D. Candidate, School of Civil and Environmental Engineering, Univ. of New South Wales, Sydney, NSW 2052, Australia (corresponding author). E-mail: maghrebi@unsw.edu.au

[2]Professor, School of Civil and Environmental Engineering, Univ. of New South Wales, Sydney, NSW 2052, Australia; and Affiliate Researcher, National ICT Australia (NICTA), Australian Technology Park, Eveleigh, Sydney, NSW 2015, Australia.

[3]Professor, School of Computer Science Engineering, Univ. of New South Wales, Sydney, NSW 2052, Australia.

Note. This manuscript was submitted on February 26, 2014; approved on October 1, 2014; published online on November 10, 2014. Discussion period open until April 10, 2015; separate discussions must be submitted for individual papers. This paper is part of the ***Journal of Computing in Civil Engineering***, © ASCE, ISSN 0887-3801/04014117(11)/$25.00.

Literature Review

Soft computing plays a key role in solving practical problems in a wide variety of disciplines such as image processing (Huang and Chau 2008), hydrology (Taormina et al. 2012; Wu et al. 2009), data mining (Zhang and Chau 2009), environmental engineering (Muttil and Chau 2006), and construction management (Chau 2004). In resource management, finding a near-optimum solution for dispatching problems is a common issue with many disciplines in engineering such as electric power (Pal et al. 2013), logistics (Yan and Zheng 2013), economics (Bijami et al. 2013), computer science (Wang et al. 2013), biomedicine (Chen et al. 2013), transportation (Lin and Ku 2013), and construction management (Tao and Tam 2013). Dispatching is a broad area but in this paper, as aforementioned, the authors have focused solely on concrete delivery.

There is not a considerable amount of published literature on RMC. Nevertheless, in the last few years, the amount of research in this area has grown considerably. Most of the research has been devoted to mathematical modeling and heuristic approaches. The RMC can be modeled as a special vehicle-routing problem (VRP; Naso et al. 2007; Schmid, unpublished data, 2007; Chen et al. 2009). The main differences between VRP and RMC that must be taken into the account are (1) a truck can only supply concrete to one customer on each trip, whereas in VRP a truck normally can supply more than one customer, and (2) concrete cannot be hauled for a long time because fresh concrete is a perishable material. Based on these differences, a set of new constraints must be added to the original VRP formulation. In the "Methodology" section, when the structure of the method is described, all required constraints will be discussed and applied in the proposed algorithm. Several attempts have been made to model the RMC dispatching effectively (Asbach et al. 2009; Feng et al. 2004; Naso et al. 2007; Yan et al. 2008; Schmid et al. 2010; Durbin and Hoffman 2008; Maghrebi et al. 2014c, d). It has been proved that a RMC-optimization problem is a NP-hard problem (Asbach et al. 2009; Maghrebi et al. 2013a, 2014a; Yan et al. 2008). Therefore, to deal with this problem, heuristic methods have been widely used in the literature. The implementation of genetic algorithm (GA) has been

highlighted more than other heuristic methods. Garcia et al. (2002) modeled the RMC for a single depot and solved it by optimization and GA. However, their approach is not practical because some realistic constraints were relaxed and considered only small instances. Feng et al. (2004) also modeled a single-depot RMC and assumed some parameters such as loading/unloading times as fixed parameters. The instances that have been considered by them are much smaller than the instances that are used in this paper. Naso et al. (2007) modeled a more realistic RMC problem by considering multidepots and penalizing the waiting times (loading/unloading) in the objective function. They also introduced a GA that is very similar to the methods presented earlier by Garcia et al. (2002) and Feng et al. (2004). However, the instances that Naso and colleagues have tested are larger than previous research. Lu (2002) and Lu et al. (2003) developed a software package called *HKCONSIM* to deal with real RMC problems. It is mainly concerned with the discrete-event simulation (DES) tool, but in its recent versions it was coupled with heuristic solvers such as GA (Cao et al. 2004; Lu and Lam 2005; Ming and Hoi-Ching 2009), particle swarm optimization (PSO; Lu et al. 2006; Wu et al. 2005), and real global positioning system data of trucks (Lu et al. 2007), thus making it a more powerful tool. Feng and Wu (2006) and Cheng and Yan (2009) had a similar approach by integrating DES with a fast messy GA. Silva et al. (2005) compared GA with ant-colony optimization (ACO) and suggested a GA–ACO method for solving RMC problems. Pan et al. (2010) proposed an improved discrete PSO (DPSO) for solving RMC-dispatching problems and recently Srichandum and Rujirayanyong (2010) compared bee-colony optimization (BCO) and tabu search (TS) with GA in this context. Despite developments in this area, the solution structure among most introduced methods is pretty much the same, especially in the GA-based method in which the chromosome structure consists of two merged parts: the first part defines the sources of deliveries, and the second part expresses the priorities of customers. The solution structure in these techniques is quite simple and easy to understand. However, a cumbersome computing process must be completed in each iteration to check the constraints or after achieving a premature solution. In this paper, the authors introduce a robust constructive heuristic method, which is supposed to be faster and more accurate than similar methods.

In the literature, rather than GA some other approaches also have been studied that will be discussed briefly in the text that follows. Yan et al. (2008) introduced a numerical method for solving the RMC-optimization problem by cutting the solution space and incorporating the branch and bound technique and the linear programming method. Lin et al. (2010) modeled the RMC as a job-shop problem. Yan et al. (2012) used decomposition and relaxation techniques coupled with a mathematical solver to solve the problem, whereas Payr and Schmid (2009) applied variable neighborhood search to deal with RMC-optimization problems. Asbach et al. (2009) made the mathematical modeling much simpler by dividing the depots and customers into subdepots and subcustomers. They also used large-scale instances for testing their introduced large neighborhood search and decomposition methods. Recently, Maghrebi et al. (2014b) presented a method for solving RMC-dispatching problem by Benders' decomposition.

Methodology

A decision about a delivery in RMC dispatching (with hard time window) covers the following three main clauses: (1) source, (2) destination, and (3) truck. If a soft time window is accepted, then the fourth clause would be *time*. As mentioned earlier, heuristic methods in the literature achieve the solution by defining the source and prioritizing customers. For example, in GA-, ACO-, DPSO-, BCO-, or TS-based methods, the solution array consists of two equal merged parts where the length of each part is equal to the number of customers and a cell in each part belongs to a customer. This means that each customer has two cells in solution array that respectively express the source and priority of each customer. When the solution is achieved, the feasibility of the solution is checked and the infeasible solutions are allocated by *out sources* or idle resources. Although there are different versions of this approach in the literature, most steps of the introduced methods are pretty much the same. Truck allocation is one of the challenging issues in these methods, even if truck allocation is embedded in the solution such as that in Lu et al. (2006). If the heuristic approach is unable to find a feasible solution, there is not any control for the number of outsourced trucks. This issue is considered in the proposed model but is worth discussion in a separate publication.

Based on a comprehensive review of the related literature, the authors realize that there is a potential for proposing a more robust meta-heuristic method to deal with complex RMC problems. The authors believe that despite good developments in this area, a faster and more accurate method can still be introduced.

To maintain consistency throughout the formulation and the algorithm, all required parameters are defined in the "Notation List" section. The body of formulation that is used for modeling the optimization problem is very similar to the method introduced by Asbach et al. (2009). However, a new formulation is introduced for evaluating the solutions as well as for minimizing the number of trucks

$$\text{Minimize} \sum_{u}\sum_{v}\sum_{k} z_{uvk}x_{uvk} - \sum_{c}\beta_c(1-y)_c \tag{1}$$

Subject to the following:

$$\sum_{u\in u_s}\sum_{v} x_{uvk} = 1 \quad \forall\ k \in K \tag{2}$$

$$\sum_{u}\sum_{v\in v_f} x_{uvk} = 1 \quad \forall\ k \in K \tag{3}$$

$$\sum_{u} x_{uvk} - \sum_{j} x_{vjk} = 0 \quad \forall\ k \in K, \quad v \in C \cup D \tag{4}$$

$$\sum_{u\in D}\sum_{k} x_{uvk} \le 1 \quad \forall\ v \in C \tag{5}$$

$$\sum_{v\in C}\sum_{k} x_{uvk} \le 1 \quad \forall\ u \in D \tag{6}$$

$$\sum_{u\in D}\sum_{k} q(k)x_{uvk} \ge q(c)y_c \quad \forall\ c, v \in C \tag{7}$$

$$-M(1-x_{uvk}) + s_u + t_{uvk} \le w_v - w_u \quad \forall\ (u,v,k) \in E \tag{8}$$

$$M(1-x_{uvk}) + \gamma + s_u \ge w_v - w_u \quad \forall\ (u,v,k) \in E \tag{9}$$

The objective function [Eq. (1)] forces optimization to find feasible solutions for all customers and penalizes if a feasible solution for customer (c) cannot be found by applying zero to y_c. Therefore, due to the large value of $\beta_c(c)$, optimization attempts to avoid unsupplied customers. Eq. (2) ensures that a truck at the start of the

day leaves once from its base and similarly Eq. (3) necessitates return of a truck to the depot/its home at the end of the day. In reality, a truck arrives at either a depot or a customer then leaves that node after loading/unloading. This concept is called *conservation of flow* and Eq. (4) ensures this issue if $u \in C$, then ($v \in D$ and $j \in C$); however, if $u \in D$, then ($v \in C$ and $j \in D \cup v_f$). In this formulation, a depot is divided into a set of subdepots based on the number of possible loadings. Similarly, a customer is divided into a set of subcustomers according to the number of required deliveries. To simplify the text, from here on a *depot* means a subdepot, which can load a truck only at a specific time and similarly a *customer* means a subcustomer who requires a delivery only at a specific time. Therefore, Eqs. (5) and (6), respectively, certify sending a truck only to a customer and a depot only supplies a customer. Eq. (7) checks demand satisfaction for customers. Eqs. (8) and (9) are designed to control timing issues. Eq. (8) ensures that concrete will be supplied to customers within the specified time, and similarly Eq. (9) ensures that the travel time for each customer will not exceed the permitted time for delivery (Υ) because fresh concrete is a perishable material and it is not possible to haul it more than (Υ), which varies according to the type of concrete.

Among the introduced heuristic methods in the literature (such as Naso et al. 2007; Yan et al. 2008; Yan and Lai 2007), any infeasibilities in achieved solutions are mostly adjusted by outsources or idle resources. This might be a quick method but the number of unscheduled jobs based on the initial solution is a major challenge for these techniques. The authors realize that the structures of those solutions (before decoding and required adjustments) cause this problem. As discussed in the prior section, the structure of the solution in the introduced heuristic methods consists of two parts. The first part finds the best depot for each customer, and the second part provides a hint for fleet allocation by prioritizing customers. The infeasibility occurs when there is no match between the acquired priorities and available resources. In these circumstances, supplying concrete from idle/loaned resources is mainly used to reduce the infeasibilities of the initial solution.

This paper proposes a new approach for constructing the solution structure while heuristic approaches are implemented. This approach is more robust and accurate. The authors discovered that finding a solution for depots needs much more iteration than finding a solution only for truck allocation. Moreover, in heuristic methods, by increasing the number of variables, the complexity of the search is increased exponentially. This means that the chance of converging to a near-optimum solution drops by increasing the solution space. This issue is evaluated in the following section.

This paper suggests that to separate the RMC problem into two detached problems that although are solved separately, they are looking for finding a common solution. This technique consists of two one-dimensional arrays with a length of i, where i is equal to the number of customers. The first array is designed for finding a solution for the supplier depot of each customer, and the second array provides a solution for allocating a truck to each customer.

As with the discussed optimization method, instead of combining a few depots and a number of projects into the model, sets of subdepots and subcustomers are defined respectively based on the number of available loading times and number of required deliveries. In addition, although this paper focuses only on GA among the heuristic methods, the proposed approach can be applied to the other similar heuristic approaches such as ACO, DPSO, BCO, or TS. This can be done by changing only the evolutionary algorithm with the proposed structure. All the mentioned techniques in the RMC domain are dealing with discrete solution spaces. Therefore, the proposed structure for RMC-dispatching problem can be solved with different discrete evolutionary techniques. The authors do not intend to add anything new to GA (Holland 1975); however, this paper does seek to investigate decreasing the complexity of the dispatching problem by splitting the main problem and solving it incrementally. The steps of the GA method are depicted in Fig. 1.

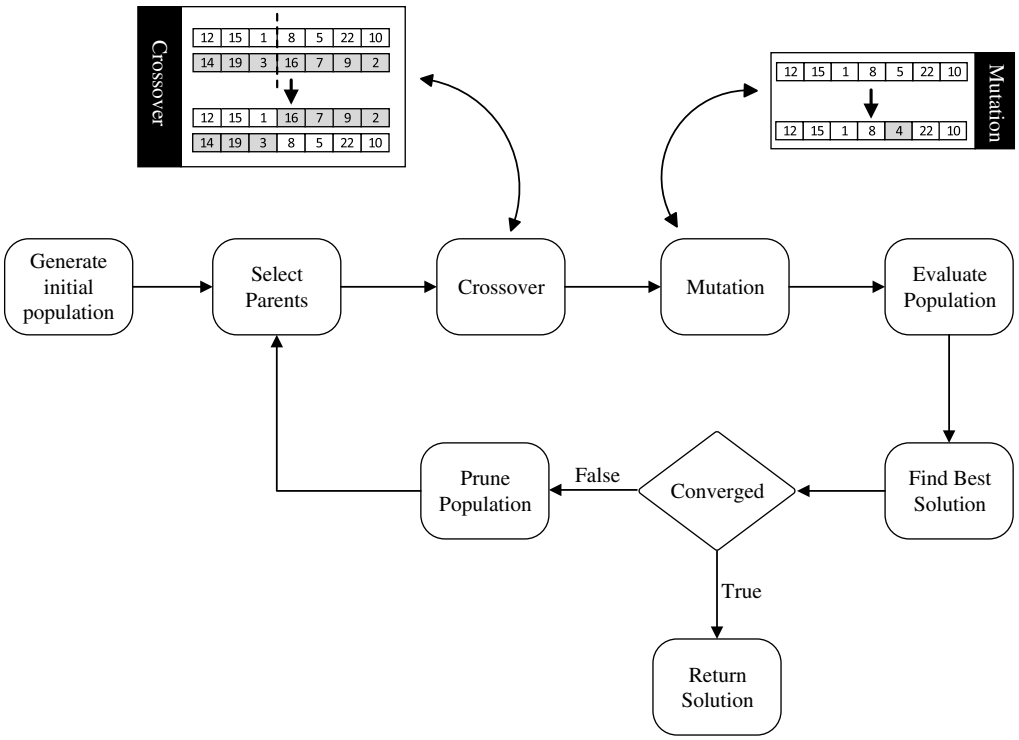

Fig. 1. Typical GA process

J. Comput. Civ. Eng.

According to this figure, generating an initial population is the first step. While this is expected to generate random solutions for an initial population, the authors modified the random process to avoid some computation tasks. This issue will be discussed in a following section. The next step is selecting a parent, which this research uses as a resampling method. This means that there is an equal chance for selected and nonselected instances in the next samples. After each selection, the selected chromosomes are put back into the population and have a chance of being selected again. The next step is crossover, which is also called *mating*. This step aims to find a better solution from more than one chromosome and the possibility of finding a better solution comes by the combination of at least two chromosomes. One-point crossover (Poli and Langdon 1998) is used in this research, which randomly selects a crossover point within two chromosomes and then interchanges their parts. The following task is mutation, which provides an opportunity for one or more genes to change their values. This diversity can lead to a faster convergence (Srinivas and Patnaik 1994). Crossover and mutation lead to producing a new generation, which is evaluated in the next stage. Based on an assessment that is conducted in this section, the chromosomes are sorted and the best solution is selected. If the achieved solution satisfies the threshold, the process is terminated; otherwise, the generation is pruned by maintaining the top chromosomes and eliminating the weak chromosomes. The process is repeated by new generation until the threshold is satisfied. In this paper, convergence between the best solutions among generations is used as a threshold. Two issues are critical when it comes to the expected solutions, namely, feasibility and reduced cost. The factor of infeasibilities in the solution is taken into account by applying a large penalty for each impractical allocation. This forces GA to avoid infeasible solutions, which is also the first strategy. This leads to a rapid pruning of very weak solutions from the population. Then, when the impact of infeasible solutions has vanished, GA iteratively must look for better solutions from the available generation. In such heuristic techniques, there is no proof of how much the achieved results are optimum. Therefore, this research intends to let GA run until there is no significant change in its performance over a certain number of iterations.

Tactical Solution

For the first separated problem, which is finding a near-optimum solution for depot allocation, Fig. 2 has been designed. This provides a list of i depots for i customers. According to this method, it is not necessary to check the feasibility of the solution for Eqs. (4) and (5). These constraints attempt to ensure that a depot is used once and a depot is allocated to each customer. Although the algorithm does not violate any of these techniques in the initial population, these constraints must be checked after crossover and mutation. After the crossover, there is 5% chance for mutation and this can be applied to each generation among the newly generated solutions. Eqs. (8) and (9) play a key role in these phases. Their job is controlling the timing issues. The demand satisfaction [Eq. (4)] is not directly checked in this phase; however, it is indirectly taken into account by applying a penalty for infeasible allocations in the fitness function. This will be considered in the next section when fleet allocation is described. For evaluating the solutions, a fitness function, which is shown in Eq. (10), has been designed. This assesses two parameters, namely, cost and infeasibilities. For cost, it calculates the sum of distances between allocated depots and customers. The second clause of Eq. (10) calculates the number of infeasible allocations; multiplying this value with a big constant (M) forces the model to avoid unsupplied customers. The iteration in this stage is stopped when the solution is converged at a rate equal to or lower than 0.001 between the last $(2 \times i)$ generations and also the number of infeasibilities in the last $(2 \times i)$ solutions is equal to zero. The process in $i \times 500$ iterations is terminated if both mentioned criteria do not satisfy

$$\sum_i \mathrm{Dis}(DA_i, i) + \sum_i (1 - \mathrm{DVI}_i)M \tag{10}$$

Operational Solution

In this stage, the authors try to find a doable truck allocation for achieved depot allocation from the preceding stage. The algorithm that is used in this stage is very similar to Fig. 2, with the exception that a truck can be allocated to more than one customer. Therefore, if k is the number of available trucks, then each gene can accept a random value between 0 and k. As with the previous stage, this structure requires fewer controlling efforts. For the remaining constraints that are not associated in the solution, Eqs. (2)–(4) do not need to be checked because the proposed solution for truck allocation covers them. According to the available database, each truck must go back to its base at end of the day, so already Eq. (3) does not need to be considered. The trucks' starting points [Eq. (2)] also need not be considered because starting points are depots and a depot is only allocated to a customer and subsequently a truck is allocated to a customer. Similarly, this applies for Eq. (4), which controls the conservation flow. This is the most crucial part in this method. Therefore, rather than Eq. (8), which is very straightforward, a combination of Eqs. (4) and (8) must be considered simultaneously. Possibly, the solution is that a set of customers is allocated to each truck. Then the feasibility of this allocation must be checked through all the customers and the allocated depots for each truck. For example, Truck number 1 is allocated to C_1, C_5, and C_{10} and, respectively, D_{10}, D_{33}, and D_{50} are assigned depots for supplying concrete to C_1, C_5, and C_{10}. In this stage, rather than checking the timing constraints between $(D_{10} - C_1)$, $(D_{33} - C_5)$, and $(D_{50} - C_{10})$, the arcs between $(C_1 - D_{33})$ and $(C_5 - D_{50})$ also must be controlled when the trucks are unloaded at a customer and travel to a depot for loading. The fitness function of this stage is designed as Eq. (11) for evaluating all populations. Rather than feasibility and cost, it controls the number of used trucks as well. The first clause calculates the distances that trucks travel between depots and customers. The second clause incorporates the traveled paths between customers after unloading and succeeding depots. The third clause takes the infeasibilities into account, whereas the fourth clause considers the number of used trucks. Both of the two last parts of this equation exert penalties with the difference that M is much larger than N, and N is much larger than the sum of the traveled distances. The originally acquired solution must be feasible and able to be used with the least cost. Therefore, GA tries to find a practical solution and then attempts to cut the costs. In this paper, the indirect costs of a truck are considered for defining the value of N. This forces GA to find a solution with fewer trucks. The threshold for this part is exactly same as the threshold used for (algorithm number 1)

$$\sum_k \sum_i \mathrm{Dis}_k(DA_i, i) + \sum_k \sum_1^{P_k - 1} \mathrm{Dis}_k[S_k(P_k), DA_{S_{k(P_k+1)}}] + \sum_i (1 - \mathrm{TVI}_i)M + \sum_k (1 - \mathrm{TC}_k)N \tag{11}$$

Algorithm 1: DEPOT Allocation

$i \leftarrow number\ of\ Customers$
$b \leftarrow number\ of\ Depots$
$n \leftarrow number\ of\ Population$
$bestsolution(1) \leftarrow a\ big\ constant$
$Threshold \leftarrow 0.001$
$Check \leftarrow 0$
$t \leftarrow 1$
AvailableDepots=$\{1, 2, ..., b\}$
`//Create empty matrices with n rows and i columns`
Generations=int[n][i]
NewGeneration=int[n][i]
`//Create empty matrices with 1 rows and i columns`
OfSp1=int[1][i]
OfSp2=int[1][i]
`//Generate initial population`
for $l \leftarrow 1$ ***to*** n **do**
    $RI \leftarrow shuffle\{1, 2, ..., b\}$
    for $p \leftarrow 1$ ***to*** i **do**
        $Generation(l, p) \leftarrow AvailableDepots(RI(p))$
`//Replenish population`
while *Check=0* **do**
    for $f \leftarrow 1$ ***to*** $n/2$ **do**
        $Parent1 \leftarrow Generation((random(1 : n)), 1 : i)$
        $Parent1 \leftarrow Generation((random(1 : n)), 1 : i)$
        $\{OfSp1, OfSp2\} \leftarrow Apply\ CrossOver(Parent2, Parent2)$
        $OfSp1 \leftarrow Apply\ Mutation(OfSp1)$
        $OfSp2 \leftarrow Apply\ Mutation(OfSp2)$
        $NewGeneration \leftarrow NewGeneration + OfSp1 + OfSp2$
    Evaluate NewGeneration using (equation 10)
    $SOL \leftarrow Find\ the\ superior\ solution\ among\ NewGeneration$
    $VIOL \leftarrow number\ of\ infeasibilities\ of\ SOL$
    if $SOL < bestsolution(t)$ **then**
        $bestsolution(t) \leftarrow SOL$
        $solutionViol(t) \leftarrow VIOL$
    else
        $bestsolution(t) \leftarrow bestsolution(t-1)$
        $solutionViol(t) \leftarrow solutionViol(t-1)$
    $Generation \leftarrow NewGeneration$
    if $t > (1 + 2 \times i)$ **then**

$$a = \frac{\left(\sum_{(k=t-(2\times i))}^{t1} bestsolution(k)\right) - ((2 \times i) \times bestsolution(t))}{(2 \times i) \times bestsolution(t)}$$

$$b = \frac{\left(\sum_{(k=t-(2\times i))}^{t} solutionViol(k)\right)}{(2 \times i)}$$

    if $(a \leq Threshold\ \ AND\ \ b = zero)\ \ OR\ \ (t > (500 \times i))$ **then**
        $Check \leftarrow 1$
    Prune the population
    $t \leftarrow t + 1$
return *bestsolution(t)*

Fig. 2. Algorithm of depot allocation

Table 1. Achieved Results

		Sequential-GA							Robust-GA						Random-solution						
		Depot allocation		Truck allocation											Depot allocation		Truck allocation				
Instance ID	ND	NG	NUAD	NG	NLTR	NTR	ET	Cost	NG	NUAD	NLTR	NTR	ET	Cost	NG	NUAD	NG	NLTR	NTR	ET	Cost
1	19	180	0	40	0	8	39	196.9	1,085	0	0	8	146	224.1	20,000	12	1,000	0	18	12	438.5
2	24	279	0	72	0	6	52	174.7	145	0	0	13	88	247.6	20,000	16	1,000	0	22	16	502.4
3	25	259	0	48	0	8	54	244.4	257	0	0	15	65	298.6	20,000	17	1,000	0	22	17	473.2
4	43	517	0	162	0	11	99	420.1	589	0	0	15	2,371	551.2	20,000	34	1,000	0	39	34	910.2
5	64	1,482	0	233	0	18	349	595.1	696	0	0	30	354	875.5	20,000	53	1,000	0	55	53	1,461.2
6	66	1,413	0	227	0	16	259	552.8	389	0	0	28	351	831.8	20,000	55	1,000	1	53	56	1,455.3
7	88	2,334	0	292	0	24	671	923.2	1,295	0	0	33	696	1,132.4	20,000	79	1,000	8	65	87	2,174.1
8	97	1,957	0	301	0	25	493	813	988	0	0	39	909	1,020.5	20,000	85	1,000	6	71	91	2,404.1
9	100	1,928	0	262	0	29	470	866.9	1,220	0	0	24	16,024	886.4	20,000	86	1,000	6	71	92	2,230.6
10	101	3,596	0	308	0	27	2,350	1,049.9	951	0	0	26	14,460	1,099.4	20,000	88	1,000	4	70	92	2,472.7
11	106	1,918	0	303	0	27	468	1,365.6	6,745	0	0	28	1,983	1,435.7	20,000	92	1,000	8	74	100	2,361.6
12	113	2,033	0	431	0	25	671	1,006.7	1,568	0	0	28	3,707	1,028.9	20,000	99	1,000	16	76	115	2,559
13	118	2,242	0	460	0	26	746	966.1	3,836	0	0	29	19,958	951.7	20,000	103	1,000	12	77	115	2,654.4
14	119	2,826	0	335	0	32	718	1,019.4	4,646	0	0	38	10,537	1,156	20,000	105	1,000	11	74	116	3,078.6
15	126	3,787	0	555	0	33	886	1,056.4	58,432	0	0	25	21,898	1,050.1	20,000	111	1,000	18	78	129	2,960.8
16	126	3,052	0	538	0	31	3,164	1,209.5	2,094	0	0	33	18,123	11,930	20,000	111	1,000	12	79	123	3,126.6
17	127	4,529	0	440	0	35	2,164	1,319	2,743	0	0	27	23,443	1,323.8	20,000	113	1,000	11	73	124	3,280.1
18	128	2,551	0	380	0	34	859	1,155.5	2,435	0	0	41	2,566	1,269	20,000	112	1,000	13	79	125	3,130.5
19	131	4,781	0	564	0	33	968	1,085.9	8,436	0	0	38	24,618	1,160.7	20,000	116	1,000	15	73	131	3,085.8
20	134	3,486	0	468	0	33	4,667	1,378.4	23,231	0	0	31	32,230	1,365.3	20,000	119	1,000	21	84	140	3,167.1
21	136	4,038	0	482	0	36	1,395	1,424.5	13,854	0	0	33	27,752	1,531.2	20,000	119	1,000	18	81	137	3,019
22	144	3,258	0	461	0	39	1,544	1,363	14,292	0	0	33	39,373	1,374	20,000	128	1,000	18	78	146	3,238.2
23	147	6,517	0	516	0	35	1,289	1,291.1	2,383	0	0	50	2,256	1,608.2	20,000	132	1,000	26	80	158	3,554.5
24	153	9,568	0	525	0	43	2,242	1,436.9	4,382	0	0	44	16,845	1,598.6	20,000	136	1,000	26	86	162	3,667.5
25	154	3,189	0	574	0	37	1,150	1,488.7	2,435	0	0	41	25,600	1,269	20,000	137	1,000	22	84	159	3,589.2
26	184	18,400	1	923	0	47	8,671	2,054.3	92,000	4	0	43	89,230	2,041.4	20,000	165	1,000	40	93	205	4,010.7
27	198	5,669	0	562	0	47	5,245	2,313.6	68,714	0	0	44	68,678	2,030	20,000	178	1,000	41	90	219	4,436.1
Average	110	3,548	~0	387	0	28	1,544	1,065.6	11,846	~0	0	31	17,195	1,529.3	20,000	96	1,000	13	68	109	2,572

Note: Cost = traveled distance by trucks (km); ET = elapsed time (s); ND = number of deliveries per instance; NG = number of generations; NLTR = number of loaned trucks; NTR = number of used trucks; and NUAD = number of unassigned deliveries.

Results and Discussion

In this section, the proposed method is tested with a real database. The achieved results are then compared against a random solution and a GA method (Maghrebi et al. 2013b), which acquires a solution in a one-piece chromosome. Among the introduced GA methods in this context, the method that was introduced by Maghrebi et al. (2013b) requires less computing effort and less concern about infeasibility because it avoids some obvious impractical allocations in initial generations. As discussed before, other introduced GA techniques in this area produce a premature solution and then try to fix the infeasibilities by defined iterative algorithms.

The authors have tried to compare the proposed method with the state-of-the-art in this area and random solution. The random solution over a large number of iterations shows the level of complexity of the problem and reveals the chances of randomly acquiring a feasible solution. The structure of random solution is exactly same as the proposed method with the difference that there is not any crossover/mutation/pruning process and only after a random generation best solution is selected and compared with the best achieved random solution. The number of generations for depot allocation is 20,000 and for truck allocation is 1,000, which are much bigger than the maximum number conducted in the proposed method.

The selected data set belongs to a branch of a RMC in Adelaide (Australia). They have four active batch plants and around 50 trucks in this area. The data set covers a month and 27 working days, which means the authors have picked 27 instances. The minimum and maximum numbers of deliveries in a day are 19 and 198, respectively. In more than 70% of instances, it is necessary to send more than 100 trucks. In addition, the demand on only 4 days is fewer than 50 trucks. A preprocessing process has been done to clean the available data set and make sure that there is no missed value or duplication in the selected instances.

The Leonardi cluster in the Faculty of Engineering at the University of New South Wales is used to facilitate the computing process and provide more accurate comparison especially for *elapsed time*. Each run uses eight processors (64 bit AMD 6174, 2.20 GHz) of a node supported with 4-GB RAM in total. *MATLAB* (*Linux* version) was used for coding all methodologies and the same

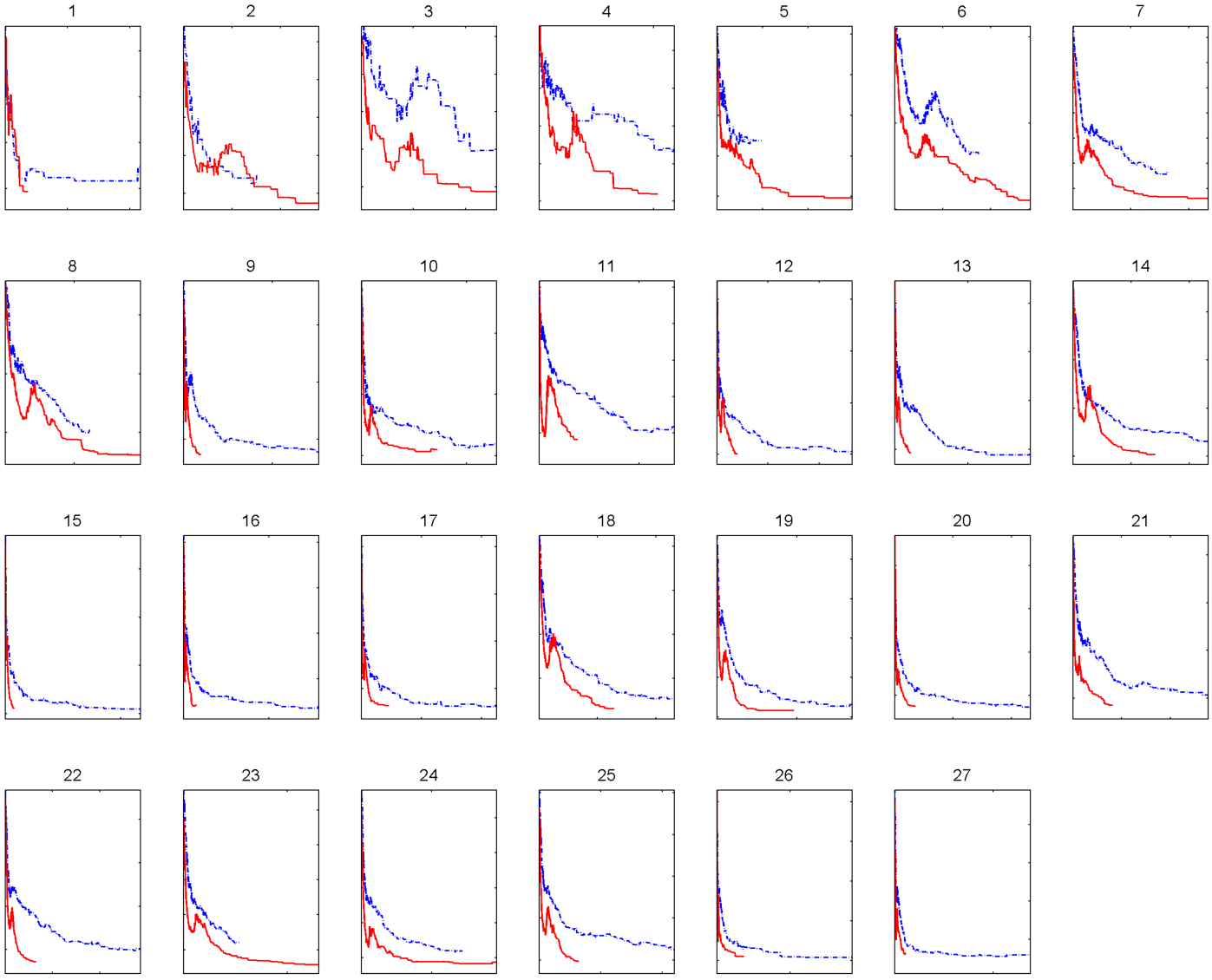

Fig. 3. Comparison of the sequential GA (solid line) and traditional GA (dashed line) in terms of cost in the best solution over generations. The *X* axis is number of generations and the *Y* axis is cost

threshold is defined for both sequential GA and traditional GA. The available instances are sorted according to their number of deliveries and then receive an ID number between 1 and 27. The results are compared according to the following metrics: cost, number of used trucks, number of required iterations, number of outsourcing in solution, and elapsed time. For calculating the cost, Eq. (12) is used and penalty functions are not associated with the cost because incorporated parameters in the penalty function are considered separately as detached metrics. The proposed GA is called *sequential GA* and in a following section will be compared against the conventional GA called *traditional GA*

$$\sum_k \sum_i \mathrm{Dis}_k(DA_i, i) + \sum_k \sum_1^{P_k-1} \mathrm{Dis}_k[S_k(P_k), DA_{S_{k(P_k+1)}}] \quad (12)$$

The achieved results for all approaches are embedded in Table 1. To achieve an unbiased comparison and considering the randomness factor, only the overall trend will be considered rather than a pairwise comparison between instances.

The main purpose of this paper is to find a solution for RMC dispatching in two stages through a tailored GA approach. As expected, this leads to a significant drop in the number of generations for the GA search. The average number of generations (NG) in sequential GA is approximately one third of the traditional GA, which subsequently speeds up the computation process by more than 10 times. As predicted, in sequential GA, the number of generations for truck allocation is much less than the number of required generations for depot allocations, which shows the complexity of each separate subproblem as well. This discrepancy is the main challenge for ordinary GA methods including traditional GA. Because the chance of getting a near-optimum solution for depot allocation in sequential GA is higher than traditional GA, both depot and trucks solutions need to be looked at simultaneously. In traditional GA, the population may include solutions that are good only for depot allocation or only for truck allocation; however, it is very difficult to define a fitness function that is able to maintain both types of mentioned solutions in the population and also increase the chance of an exchange between them.

Despite an improvement in the speed of the computing process when sequential GA is used, the quality of the solutions is

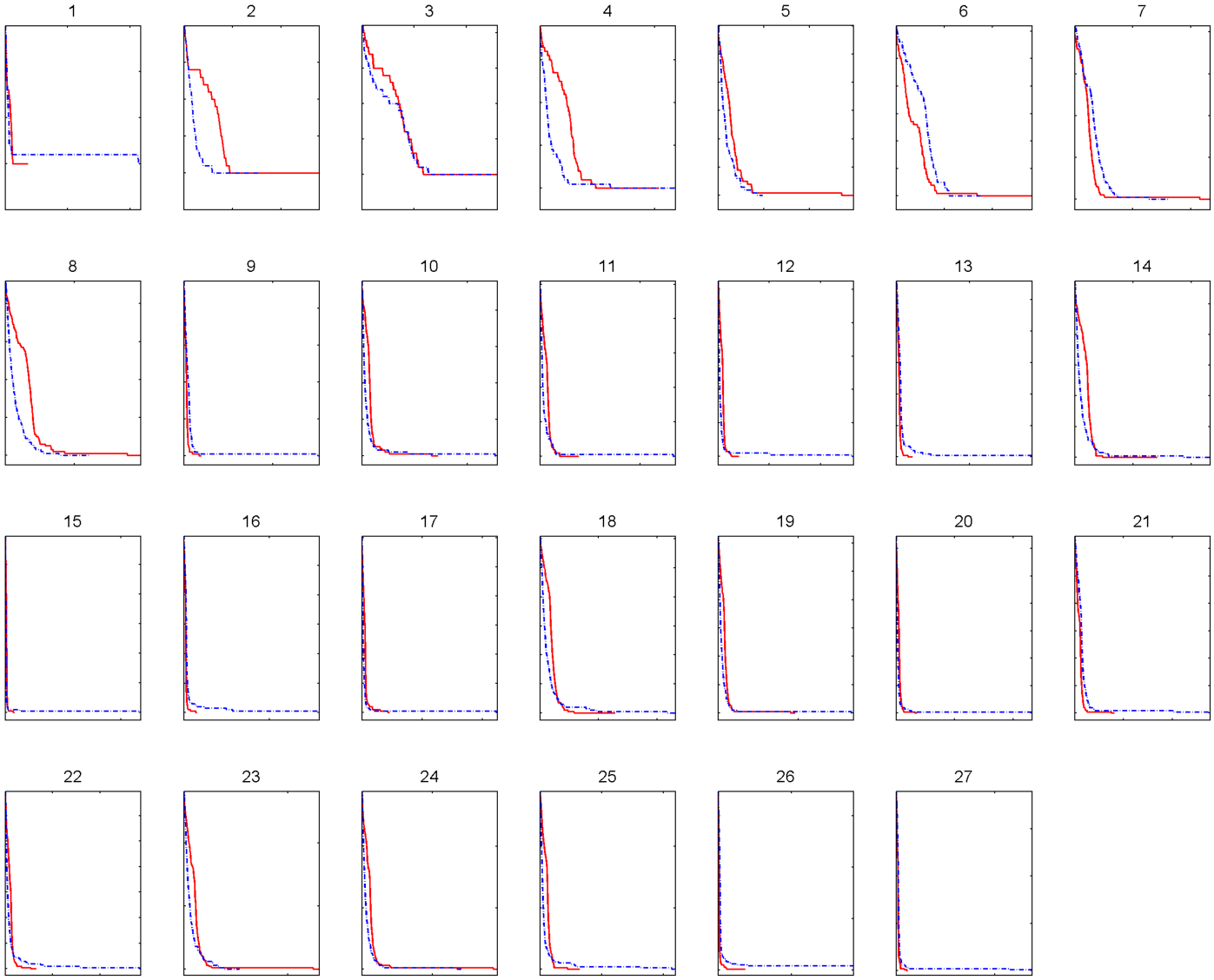

Fig. 4. Comparison of the sequential GA (solid line) and traditional GA (dashed line) in terms of number of infeasibilities in the best solution over generations. The X axis is number of generations and Y axis is number of infeasibilities

J. Comput. Civ. Eng.

increased as well. For assessing the quality of solutions, three metrics are used; the first is the number of outsourcings in the solution. Two scenarios might happen if the selected approach cannot find a feasible solution such as those marked in Table 1 by NUAD and NLTR, which respectively refer to number of unassigned deliveries and number of loaned trucks. The infeasibility in the solution might occur when a feasible depot allocation for all deliveries cannot be found and unassigned customers must be supplied from outsourced depots. The NUAD considers this concern. The second possibility is of infeasibility in the solution due to an unavailability of trucks that NLTR maintains, which causes apprehension. This parameter shows how many trucks are loaned for each instance. The NLTR for both sequential GA and traditional GA in the entire instance is zero. In addition, NUAD in most of the instances for both techniques is zero with the exception of instance 26. Although there cannot be a solid judgment based on only one instance, this instance has the highest number of generations in both techniques, which shows the complexity of this instance. This also supports the issue discussed earlier that the chance of finding a feasible solution in sequential GA is higher than traditional GA when sequential GA is converged to a more practical allocation with lesser generation. Nevertheless, NUAD and NLTR indicate that the performance of both sequential GA and traditional GA in terms of acquiring feasible solutions is acceptable.

The number of used trucks (NTR) is the next metric, which is applied for assessing the quality of solutions. Although sequential GA achieves a slightly better result and is able to supply concrete with fewer trucks, it is not a significant difference, likely because traditional GA is capable of providing a good enough solution for truck allocation while simultaneously looking for a proper depot allocation.

The last used metric is cost, which covers all traveled distances by trucks. For RMC owners, this parameter is very important because it is a main component of their operational costs. In terms of traveled distance only, sequential GA supplies the orders at around 30% less cost in comparison with traditional GA. In summary, the achieved results show that sequential GA can solve RMC-dispatching problems 10 times faster than traditional GA with a 30% reduction in cost for the field data set analyzed in this research.

This research now intends to study the behavior of GA techniques in more detail by focusing on improvements in solutions over generations. Fig. 3 illustrates the cost of best achieved solutions over generations for all 27 instances. The number above each figure represents the ID of the instance. The solid and dashed lines, respectively, reflect the behavior of sequential GA and traditional GA. Numbers are not depicted in axis due to the density of the figure; however, the information provided in Table 1 can be associated for better understanding Figs. 3 and 4. Although Eq. (10) is used for evaluating the chromosomes, the depicted lines show only the cost of the best solution in each generation without considering the penalties. The new issue that is indicated by this figure is that in most instances, sequential GA converges faster than traditional GA. However, in some instances, such as instances 2, 5, 6, 7, 8, 23, and 24, sequential GA produced more generation but led to relatively better solutions in terms of cost. In addition, in large-scale problems where the number of required deliveries is greater than 100 (instances 10–27), sequential GA tends to converge much faster than traditional GA, which shows the ability of this technique in complex problems. Similarly, Fig. 4 shows the strength of sequential GA in the reduction of infeasibilities in solutions over generations, although in some instances traditional GA converges quickly but stocked when only a few infeasibilities remain in the solution. However, in most instances, especially large-scale ones, sequential GA found a feasible solution with far fewer generations.

Sensitivity analysis to study the interaction between depots and trucks allocation and testing the presented sequential approach with other evolutionary techniques such as PSO, ACO, and TS can be considered as future works.

Conclusion

This paper introduces a sequential heuristic method for solving large-scale RMC problems. In RMC, when a time window is not allowed, two factors must be considered for each delivery, namely, source of delivery and a proper truck. In the literature, the solution structure among most of the introduced heuristic methods is relatively similar. This is especially the case in GA-based methods where the chromosome structure consists of two merged parts: the first part defines the sources of deliveries, and the second part expresses the priorities of customers. However, in this paper, a sequential heuristic method is proposed, which finds the solution in two separate stages. A tailored GA search procedure is selected for implementing the proposed approach, and a field data set is used for evaluating the proposed method. The data cover a large RMC dispatcher for a period of 1 month. The results show that sequential GA is more robust than traditional GA when it converges 10 times faster with achieved solution with a 30% reduction in cost for the data set used. The fixed time of delivery is the limitation of this and similar approaches, but a time window can be considered as future work. The challenge for RMC-dispatching problems with soft time windows is that the timing variables are nonintegers unlike depot- and truck-allocation variables. It is suggested that a hybrid heuristic method might be a proper approach for tackling this expanded problem.

Acknowledgments

The authors acknowledge computing time at the Leonardi cluster in the Faculty of Engineering, the University of New South Wales and National Information Communications Technology Australia is funded by the Australian Department of Communications and the Australian Research Council through the Information Communications Technology Centre of Excellence program.

Notation

The following symbols are used in this paper:

C = set of customers;
D = set of depots;
DA_c = allocated depot to customer c;
$\mathrm{Dis}(u, v)$ = distance between u and v;
$\mathrm{Dis}_k(u, v)$ = distance between u and v when truck k is allocated otherwise 0;
DVI_i = 1 if the allocated depot for customer v is infeasible otherwise 0;
K = set of vehicles;
M = big constant;
N = constant;
P_k = number of assigned customers to truck k;
$q(c)$ = demand of customer c;
$q(k)$ = maximum capacity of vehicle k;
S_k = set of assigned customer(s) to truck k;
$s(u)$ = service time at the depot u;

TC_k = 0 if at least one customer was assigned to truck k otherwise 1;
TVI_v = 1 if the allocated truck for customer v is infeasible otherwise 0;
$t(u, v, k)$ = travel time between u and v with vehicle k;
U_s = set of starting points;
V_f = set of ending points;
W_o = time at location o;
x_{uvk} = 1 if route between u and v with vehicle k is selected, 0 otherwise;
y_c = 1 if total demand of customer c is supplied, 0 otherwise;
$z(u, v, k)$ = cost of travel between u and v with vehicle k;
$\beta_c(c)$ = penalty of unsatisfying the customer c; and
Υ = maximum time that concrete can be hauled.

References

Asbach, L., Dorndorf, U., and Pesch, E. (2009). "Analysis, modeling and solution of the concrete delivery problem." *Eur. J. Oper. Res.*, 193(3), 820–835.

Bijami, E., Jadidoleslam, M., Ebrahimi, A., Askari, J., and Farsangi, M. M. (2013). "Implementation of imperialist competitive algorithm to solve non-convex economic dispatch problem." *J. Chin. Inst. Eng.*, 37(2), 232–242.

Cao, M., Lu, M., and Zhang, J.-P. (2004). "Concrete plant operations optimization using combined simulation and genetic algorithms." *Proc., 2004 Int. Conf. on Machine Learning and Cybernetics*, IEEE, New York, 4204–4209.

Chau, K. W. (2004). "A two-stage dynamic model on allocation of construction facilities with genetic algorithm." *Autom. Constr.*, 13(4), 481–490.

Chen, H.-K., Hsueh, C.-F., and Chang, M.-S. (2009). "Production scheduling and vehicle routing with time windows for perishable food products." *Comput. Oper. Res.*, 36(7), 2311–2319.

Chen, J.-L., Tsai, C.-W., Chiang, M.-C., and Yang, C.-S. (2013). "A high performance cloud-based protein-ligand docking prediction algorithm." *BioMed Res. Int.*, 2013, 1–8.

Cheng, T. M., and Yan, R. Z. (2009). "Integrating messy genetic algorithms and simulation to optimize resource utilization." *Comput.-Aided Civ. Infrastruct. Eng.*, 24(6), 401–415.

Durbin, M., and Hoffman, K. (2008). "OR PRACTICE—The dance of the thirty-ton trucks: Dispatching and scheduling in a dynamic environment." *Oper. Res.*, 56(1), 3–19.

Feng, C.-W., Cheng, T.-M., and Wu, H.-T. (2004). "Optimizing the schedule of dispatching RMC trucks through genetic algorithms." *Autom. Constr.*, 13(3), 327–340.

Feng, C.-W., and Wu, H.-T. (2006). "Integrating fmGA and CYCLONE to optimize the schedule of dispatching RMC trucks." *Autom. Constr.*, 15(2), 186–199.

Garcia, J., Lozano, S., Smith, K., Kwok, T., and Villa, G. (2002). "Coordinated scheduling of production and delivery from multiple plants and with time windows using genetic algorithms." *Proc., 9th Int. Conf. on Neural Information Processing, 2002 (ICONIP'02)*, IEEE, New York, 1153–1158.

Holland, J. H. (1975). *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence*, University of Michigan Press, Ann Arbor, MI.

Huang, Z.-K., and Chau, K.-W. (2008). "A new image thresholding method based on Gaussian mixture model." *Appl. Math. Comput.*, 205(2), 899–907.

Lin, D. Y., and Ku, Y. H. (2013). "Using genetic algorithms to optimize stopping patterns for passenger rail transportation." *Comput.-Aided Civ. Infrastruct. Eng.*, 29(4), 264–278.

Lin, P.-C., Wang, J., Huang, S.-H., and Wang, Y.-T. (2010). "Dispatching ready mixed concrete trucks under demand postponement and weight limit regulation." *Autom. Constr.*, 19(6), 798–807.

Lu, M. (2002). "HKCONSIM: A simulation platform for planning and optimizing concrete plant operations in Hong Kong." *Proc., Int. Conf. on Innovation and Sustainable Development of Civil Engineering in the 21st Century*, 278–283.

Lu, M., Anson, M., Tang, S., and Ying, Y. (2003). "HKCONSIM: A practical simulation solution to planning concrete plant operations in Hong Kong." *J. Constr. Eng. Manage.*, 10.1061/(ASCE)0733-9364(2003)129:5(547), 547–554.

Lu, M., Dai, F., and Chen, W. (2007). "Real-time decision support for planning concrete plant operations enabled by integrating vehicle tracking technology, simulation, and optimization algorithms." *Can. J. Civ. Eng.*, 34(8), 912–922.

Lu, M., and Lam, H.-C. (2005). "Optimized concrete delivery scheduling using combined simulation and genetic algorithms." *2005 Proc., 37th Winter Simulation Conf.*, Association for Computing Machinery, New York, 2572–2580.

Lu, M., Wu, D.-P., and Zhang, J.-P. (2006). "A particle swarm optimization-based approach to tackling simulation optimization of stochastic, large-scale and complex systems." *Advances in machine learning and cybernetics*, D. S. Yeung, Z.-Q. Liu, X.-Z. Wang, and H. Yan, eds., Vol. 3930, Springer, Berlin, 528–537.

Maghrebi, M., Periaraj, V., Waller, S. T., and Sammut, C. (2014a). "Solving ready-mixed concrete delivery problems: Evolutionary comparison between column generation and robust genetic algorithm." *Comput. Civ. Build. Eng.*, 680–688.

Maghrebi, M., Periaraj, V., Waller, S. T., and Sammut, C. (2014b). "Using benders decomposition for solving ready mixed concrete dispatching problems." *31st Int. Symp. on Automation and Robotics in Construction and Mining*, Q. Ha, X. Shen, and A. Akbarnezhad, eds., Univ. of Technology, Sydney, Sydney, NSW, Australia.

Maghrebi, M., Rey, D., Waller, S. T., and Sammut, C. (2014c). "Reducing the number of decision variables in ready mixed concrete for optimally solving small instances in a practical time." *CSCE 2014 General Conf.*, Canadian Society for Civil Engineering, Halifax, U.K.

Maghrebi, M., Waller, S. T., and Sammut, C. (2013a). "Reconstruction of an expert's decision making expertise in concrete dispatching by machine learning." *J. Civ. Eng. Archit.*, 7(12), 1540–1547.

Maghrebi, M., Waller, S. T., and Sammut, C. (2013b). "Scheduling concrete delivery problems by a robust meta heuristic method." *2013 7th UKSim/AMSS European Symp. on Computer Modeling and Simulation (EMS)*, IEEE, Manchester, U.K., 354–359.

Maghrebi, M., Waller, S. T., and Sammut, C. (2014d). "Assessing the accuracy of expert-based decisions in dispatching ready mixed concrete." *J. Constr. Eng. Manage.*, 10.1061/(ASCE)CO.1943-7862.0000853, 06014004.

MATLAB version 8.2.0.701 [Computer software]. Natick, MA, MathWorks.

Ming, L., and Hoi-Ching, L. (2009). "Simulation-optimization integrated approach to planning ready mixed concrete production and delivery: Validation and applications." *2009 Proc., Winter Simulation Conf.*, IEEE, New York, 2593–2604.

Muttil, N., and Chau, K.-W. (2006). "Neural network and genetic programming for modelling coastal algal blooms." *Int. J. Environ. Pollut.*, 28(3/4), 223–238.

Naso, D., Surico, M., Turchiano, B., and Kaymak, U. (2007). "Genetic algorithms for supply-chain scheduling: A case study in the distribution of ready-mixed concrete." *Eur. J. Oper. Res.*, 177(3), 2069–2099.

Pal, B. B., Mukhopadhyay, A., Biswas, P., and Mukherjee, S. (2013). "Using fuzzy goal programming to solve electric power generation and dispatch problem via genetic algorithm." ⟨http://csidl.org/handle/123456789/454⟩.

Pan, L., Liya, W., Xihai, D., and Xiang, G. (2010). "Scheduling of dispatching ready mixed concrete trucks trough discrete particle swarm optimization." *2010 IEEE Int. Conf. on Systems Man and Cybernetics (SMC)*, IEEE, New York, 4086–4090.

Payr, F., and Schmid, V. (2009). "Optimizing deliveries of ready-mixed concrete." *2nd Int., Conf. on Logistics and Industrial Informatics, 2009 (LINDI 2009)*, IEEE, New York, 1–6.

Poli, R., and Langdon, W. B. (1998). "Schema theory for genetic programming with one-point crossover and point mutation." *Evol. Comput.*, 6(3), 231–252.

Schmid, V., Doerner, K. F., Hartl, R. F., and Salazar-González, J.-J. (2010). "Hybridization of very large neighborhood search for ready-mixed concrete delivery problems." *Comput. Oper. Res.*, 37(3), 559–574.

Silva, C. A., Faria, J. M., Abrantes, P., Sousa, J. M. C., Surico, M., and Naso, D. (2005). "Concrete delivery using a combination of GA and ACO." *44th IEEE Conf. on Decision and Control, 2005 and 2005 European Control Conf. CDC-ECC '05*, IEEE, New York, 7633–7638.

Srichandum, S., and Rujirayanyong, T. (2010). "Production scheduling for dispatching ready mixed concrete trucks using bee colony optimization." *Am. J. Eng. Appl. Sci.*, 3(1), 7–14.

Srinivas, M., and Patnaik, L. M. (1994). "Adaptive probabilities of crossover and mutation in genetic algorithms." *IEEE Trans. Syst. Man Cybern.*, 24(4), 656–667.

Tao, R., and Tam, C.-M. (2013). "System reliability theory based multiple-objective optimization model for construction projects." *Autom. Constr.*, 31, 54–64.

Taormina, R., Chau, K.-W., and Sethi, R. (2012). "Artificial neural network simulation of hourly groundwater levels in a coastal aquifer system of the Venice lagoon." *Eng. Appl. Artif. Intell.*, 25(8), 1670–1676.

Wang, Y., Lin, X., and Pedram, M. (2013). "A nested two stage game-based optimization framework in mobile cloud computing system." *2013 IEEE 7th Int. Symp. on Service Oriented System Engineering (SOSE)*, IEEE, New York, 494–502.

Wu, C. L., Chau, K. W., and Li, Y. S. (2009). "Predicting monthly streamflow using data-driven models coupled with data-preprocessing techniques." *Water Resour. Res.*, 45(8), W08432.

Wu, D.-P., Lu, M., and Zhang, J.-P. (2005). "Efficient optimization procedures for stochastic simulation systems." *Proc., 2005 Int. Conf. on Machine Learning and Cybernetics*, IEEE, New York, 2895–2900.

Yan, N. N., and Zheng, D. (2013). "A study on the agent-based vehicles dispatching optimization at container terminals." *Appl. Mech. Mater.*, 241, 1745–1750.

Yan, S., and Lai, W. (2007). "An optimal scheduling model for ready mixed concrete supply with overtime considerations." *Autom. Constr.*, 16(6), 734–744.

Yan, S., Lai, W., and Chen, M. (2008). "Production scheduling and truck dispatching of ready mixed concrete." *Transp. Res. Part E Logist. Transp. Rev.*, 44(1), 164–179.

Yan, S., Lin, H., and Jiang, X. (2012). "A planning model with a solution algorithm for ready mixed concrete production and truck dispatching under stochastic travel times." *Eng. Optim.*, 44(4), 427–447.

Zhang, J., and Chau, K.-W. (2009). "Multilayer ensemble pruning via novel multi-sub-swarm particle swarm optimization." *J. Univers. Comput. Sci.*, 15(4), 840–858.