2013 European Modelling Symposium

Scheduling Concrete Delivery Problems by a Robust Meta Heuristic Method

Mojtaba Maghrebi,Travis S. Waller
School of Civil and Environmental Enginnering
The University of New South Wales (UNSW)
Sydney, Australia
{maghrebi,s.waller}@unsw.edu.au

Claude Sammut
School of Computer Science Enginnering
The University of New South Wales (UNSW)
Sydney, Australia
claude@cse.unsw.edu.au

***Abstract*—Finding practical methods for resources allocation in Ready Mixed Concrete (RMC) is a critical issue. In the literature, heuristic methods have been mostly used for solving the RMC problem. The introduced methods are mostly intended to find a premature solution and then to try to fix the indefeasibilities by defined iterative algorithms. In this paper a new approach is proposed for constructing the solution structure when discrete heuristic methods are used. Genetic Algorithm (GA) has been selected for implementing the proposed idea, and for evaluating the model a large scale dataset is used. The dataset covers an active RMC for a period of one month. The comprehensive tests show that the proposed method is able to find a feasible solution without any need to adjust the method.**

Keywords; Ready Mixed Concrete; Resource Allocation; Heuristic

I. Introduction

Ready Mixed Concrete(RMC) problems can be categorized as specified Vehicle Routing Problems (VRP) ([1], [2], [3]).In general, a VRP itself is a kind of Travelling Salesman Problem (TSP) ([4], [5]). It has been approved that solving even a simple TSP for a large instance is computationally intractable [6], consequently this issue in more complex problems such as VRP or RMC is more challenging. Although this has been approved, VRP ([4], [7]) and RMC ([8], [1]) are characterized as classic NP-hard problems. In the literature, heuristic methods have been widely used to tackle these problems especially in the RMC domain. Most of the introduced techniques, try to attain a premature solution by employing heuristic methods and then pruning the solution. In this paper we introduce a novel approach for constructing solution while heuristic methods are supposed to be used. It is also attempted to avoid any unnecessary computing tasks by adjusting the random based procedures.

II. Related Works

Most of the publications in the RMC domain have been devoted to implementing heuristic methods which Genetic Algorithm (GA) has been highlighted more than other heuristic methods. Garcia et al. [9] modelled the RMC for a single depot and solved it via optimization and GA. However, their approach is not practical because some realistic constraints were relaxed and considered only small instances. Feng et al. [10] also modelled 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. [2] modelled a more realistic RMC problem by considering multi-depots and penalizing the waiting times (loading/unloading) in the objective function. They also introduced a GA algorithm which is very similar to the methods that were presented earlier by [9], [2] .However, the instances that Naso and colleagues have tested are larger than in the previous research. Lu [11] developed a software package called HCKCONSIM to deal with real RMC problems. This mainly concerned the discrete event simulation (DES) tool but in its recent versions was coupled with heuristic solvers such as GA ([12],[13], [14]) , Particle Swarm Optimization (PSO) ([15],[16]) and real GPS data of trucks [17] in order to make a more powerful tool. Feng and Wu [18] and Cheng and Yan [19] had a similar approach by integrating DES with a fast messy GA algorithm. Silva et al. [20] compared GA with Ant Colony Optimization (ACO) and suggested a GA-ACO method for solving RMC problems. Pan et al. [21] proposed an improved Discrete PSO (DPSO) for solving RMC dispatching problems and recently Srichandum and Rujirayanyong [22] 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 same, especially in the GA based method where the chromosome structure consists of two merged parts: the first part defines the sources of deliveries; 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 we introduce a new heuristic method which is intended to be faster and need less computing effort.

III. Methodology

As mentioned above, introduced 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 which 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 via out-sources or idle resources. Although there are different versions of this approach in the literature, most of steps in the introduced methods are pretty much same. Truck allocation is one of the challenging issues in these methods, even if truck allocation is embedded in the solution such as (Lu, Wu et al. 2006), because if the heuristic approach is unable to find a feasible solution, there is no control of the number of out-sourced trucks. Among the introduced heuristic methods in the literature such as ([2],[23],[8]), any indefeasibilities in achieved solutions are mostly adjusted by out-sources or idle resources. This might be a fast method but the number of unscheduled jobs based on the initial solution presents a major challenge for these techniques. The authors realize that the structures of those solutions (before decoding and required adjustments) create this problem. The infeasibility occurs when there is no match between the acquired priorities and available resources. Under these circumstances, supplying concrete from idle/loaned resources is mainly used to reduce the indefeasibilities of the initial solution. The concept of constructing the solution structure in a new fashion, which is introduced in this paper, was inspired from optimization where some scholars such as [1] divide depots and customers into a set of depots and customers, respectively based on the number of available loading times and the number of required deliveries. To simplify the

978-1-4799-2578-0/13 $31.00 © 2013 IEEE
DOI 10.1109/EMS.2013.64

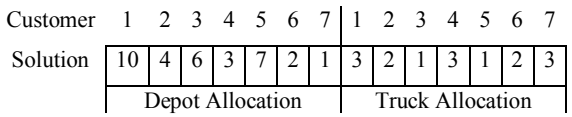

Fig. 1. Proposed structure solution for an instance with 7 customers

text, from here on a depot means a sub-depot which can load a truck only at a specific time; similarly, a customer means a sub-customer that requires a delivery only at a specific time. This method is used in this paper for manipulating the solution structure. Also, while this paper focuses only on Genetic Algorithm [24] among the heuristic methods, the proposed approach can also be applied to the other similar discrete heuristic approaches such as ACO, DPSO, BCO or TS. This can be done by changing only the evaluation algorithm while the proposed structure is used. The proposed solution structure for an RMC that supposes to supply i customers consists of a chromosome with $2 \times i$ gens. The gens 1 to i are intended to find depot allocation for customer 1 to i and gens i+1 to $2 \times i$ are dedicated to finding a proper way to allocate trucks for customer 1 to i. Fig. (1) shows the solution structure of RMC which must supply 7 deliveries. It shows that, for example, Customer#1 is supplied from Depot#10 and with Truck#3.

(Algorithm 1) is recommended for producing the initial population. The main advantage of this algorithm is avoiding infeasible solutions. Because customers that need more than one delivery were divided into a set of customers, then it is supposed to send only one truck to a customer. Similarly, because a depot was divided into a set of sub-depots, a depot can only supply fresh concrete to one customer. Correspondingly, a customer can accept only one delivery. (Algorithm 1) ensures all mentioned concerns in the initial population while conventional GA produces an initial population at random. Implementing the proposed method can lead to a faster convergence due to a cut in the number of indefeasibilities from the initial population.

Algorithm 1:

$i \leftarrow number\ of\ Customers$
$b \leftarrow number\ of\ Depots$
$n \leftarrow number\ of\ Population$
$t \leftarrow number\ of\ Trucks$
AvailableDepots=$\{1, 2, ..., b\}$
Generations=int[n][$2 \times i$]
`//Generate initial population`
for $l \leftarrow 1$ ***to*** n **do**
  $RI \leftarrow shuffle\{1, 2, ..., b\}$
  for $p \leftarrow 1$ ***to*** i **do**
    $Generation(l, p) \leftarrow AvailableDepots(RI(p))$
    $Generation(l, p + i) \leftarrow Random(1, t)$

After acquiring the initial population parents must be selected for crossover. We use the resampling method for selecting parents, which means that there is an equal chance for selected and non-selected 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 [25] is used in this research which randomly selects

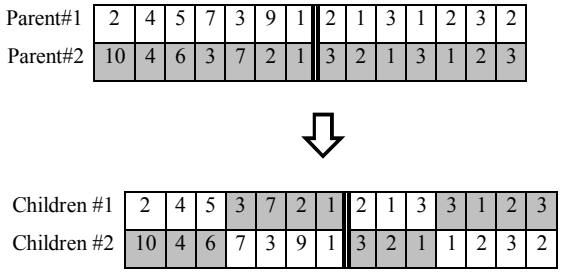

Fig. 2. Crossover for an instance of 7 customers

a crossover point within two chromosomes and then interchanges their parts. In details, cannot exchange the gens belong to the truck allocation part with gens belong to the depot allocation, the crossover is performed between the related parts of a pair of chromosomes. For each mating, the crossover point (α) is randomly chosen between 1 to i, and then children#1 are produced by merging gens 1 to from parent#1, α+1 to i from parent#2, i+1 to α+i from parent#1 and α+i+1 to 2i from parent#2. By changing the order of parent#1 and parent#2 in children#1, children#2 can be constructed. This approach avoids any exchanges between values related to the depots and values related to the trucks. Fig. 2 shows this procedure for an instance with 7 deliveries when the value of was randomly obtained.

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 [26]. After replenishing the population, all chromosomes are evaluated by a fitness function (equation. 1)

$$\sum_{1}^{k} \sum_{1}^{P_k - 1} Dis_k \left(S_k(P_k), DA_{S_{k(P_k+1)}} \right) + \sum_{1}^{i} (1 - DVI_i) \times M + \sum_{1}^{i} + (1 - TVI_i) \times M \quad (1)$$

when:

K	number of trucks
M	a large constant
DAc	allocated depot to customer c
Dis (u,v)	distance between u and v
Disk(u,v)	distance between u and v when truck k is allocated otherwise 0
DVIv	1 if the allocated depot for customer v is infeasible otherwise 0
TVIv	1 if the allocated truck for customer v is infeasible otherwise 0
TCk	0 if at least one customer was assigned to truck k otherwise 1
Sk	set of assigned customer(s) to truck k
Pk	number of assigned customers to truck k

For calculating DVIi and TVIi the timing constraints are very important. Fresh concrete is a perishable material and it is not possible to haul it more than (γ) which varies according to the type of concrete. In addition, the time between the availability of the allocated depot and the ordered time by the customer must be matched. After evaluating all the population, the chromosomes are sorted based on their fitness value 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 ones. The process is repeated by a 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: feasibility and reduced cost. The factor of

indefeasibilities 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 our initial strategy as well. This leads to a rapid pruning of very weak solutions from the population. Then, when the impact of infeasible solutions has vanished, GA must look for better solutions iteratively from the available generation. In such heuristic techniques there is no proof of how much the achieved results are optimum. Thus, we intend to allow GA to run until there is no significant change in its performance over a certain number of iterations. The whole GA process is summarized in Fig.3.

IV. Field Data

The proposed method is tested via a real database and the achieved results are compared against a 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 the 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 so far. The number of generations is 100,000 while the population size is 300. This means randomly generating 30 million solutions for each instance. The selected dataset belongs to a branch of an RMC in Adelaide (Australia). They have 4 active batch plants and around 50 trucks in this area. The dataset covers a month and 27 working days, which means we have 27 instances. The minimum and maximum numbers of deliveries in a day are respectively 19 and 198. In more than 70% of instances, it is necessary to send more than 100 trucks. Also, on four days the demand is fewer than 50 trucks. The Leonardi cluster in the Faculty of Engineering at the University of New South Wales (UNSW) is used to facilitate the computing process and provide more accurate comparisons, especially for elapsed time. Each run uses 8 processors (6174 2.20 GHz) of a node supported with 4 GB RAM in total. Matlab 2012b (Linux version) was used for coding all methodologies. The available instances are sorted according to the number of deliveries and then receive an ID number of between 1 and 27. The results are compared according to the following metrics:

- Cost
- Number of Generations (NG)
- Number of out-sourcings in the solution (NUAD)
- Elapsed time (ET)

In calculating the cost, only trucks routes are considered and penalty functions are not associated in the cost because those are considered separately as detached metrics. The proposed GA is called Robust-GA and in a following section will be compared against the random solution called Random-Solve. The achieved results for both approaches are embedded in (Table I). To achieve an unbiased comparison, and taking the randomness factor into account, only the overall trend will be considered rather than a pairwise comparison between instances.

From the results attained by Random-Solve we can deduce the low chance of achieving a feasible solution for RMC problems. In some instances, NUAD is larger than the number of customers because NUAD is calculated per gen and counts any infeasible solution regardless of whether it belongs to depot allocation or truck allocation. In an appraisal of Random-Solve, we can judge that over all instances, the chance of achieving a practical solution even for small instances is very low, which shows the complexity of the RMC problem. As expected, the proposed method is able to achieve a feasible solution for most customers without any adjustment, while a similar approach produces a premature solution and then tries to fix the indefeasibilities by defined iterative algorithms. Among all 27 instances there are 2971 customers in total; Robust-GA found a feasible solution for 2967 customers and only missed 4 customers. This shows the robustness of the proposed method in solving the

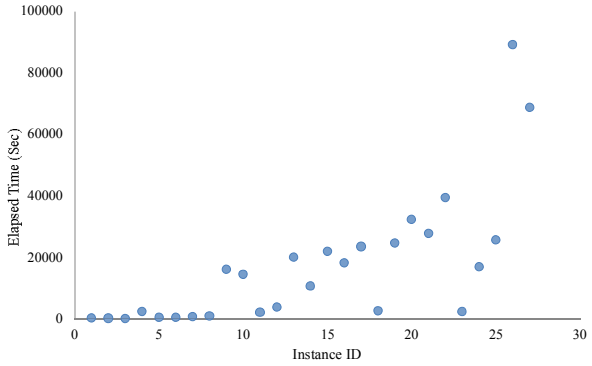

Fig. 4. Relation between size of problem and elapsed time

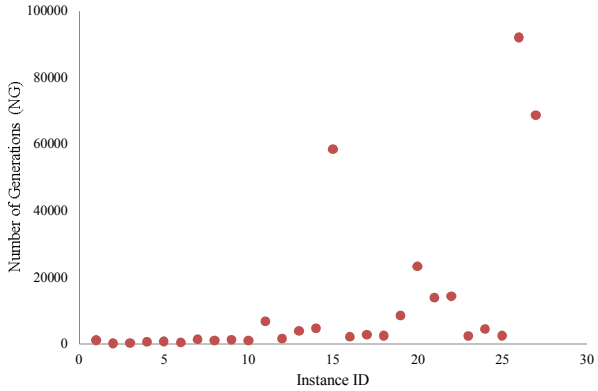

Fig. 5. Relation between size of problem and number of generations

RMC problem. Figure.4 illustrates that by increasing the size of the problem the Elapsed Time (ET) grows exponentially. Figure. 5 shows that there is not a linear relationship between computing time and the Number of Generations (NG). A small instance might need many generations but is computed quickly because its solution size is small; however, in general, it can be proved that by increasing the size of instances the chances of quickly satisfying the threshold declines and consequently needs more iterations. The last used metric is cost which covers all travelled distances by trucks. For RMC owners this parameter is very important because it is a main part of their operational costs. It also provides strong proof of how much the decisions are accurate. In general word, by increasing the size of problem, possibly the cost might be increased as well which (Table 1) support this issue. Also, in terms of only cost there is a big difference between Robust-GA and Random-Solve while the average number of infeasible solution for Robust-GA is almost zero.

V. Conclusion

This paper introduces a robust heuristic method for solving large scale Ready Mixed Concrete (RMC) problems. In RMC, when a time-window is not allowed, two factors must be considered for each delivery: source of delivery and a proper truck. In the literature, the solution structure among most of the introduced heuristic methods is pretty much same. This is especially the case in GA based method 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 new approach for constructing the structure of solution is proposed which is able to find the solution without any need to adjustment algorithms.

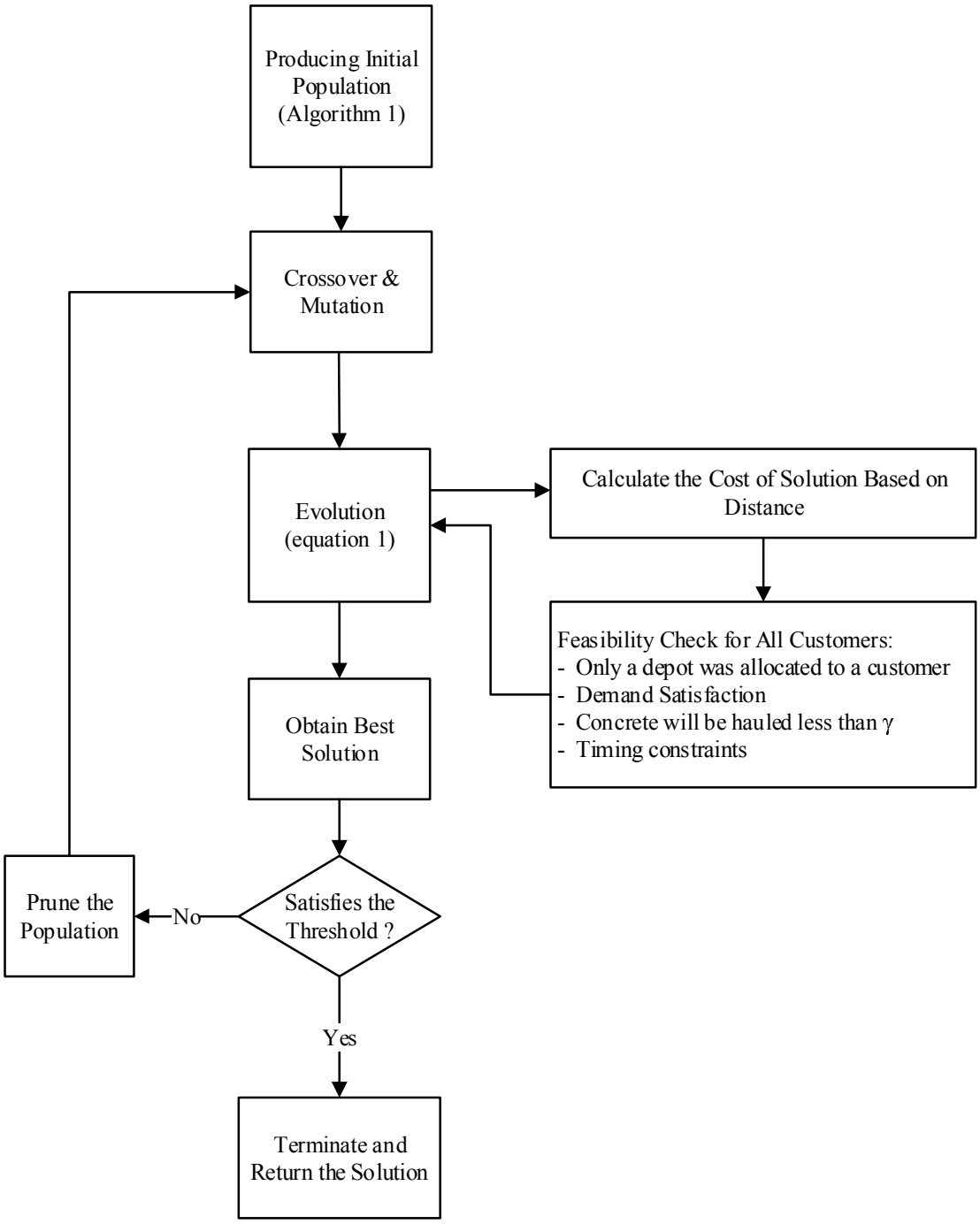

Fig. 3. The GA process

Genetic Algorithm (GA) is selected for implementing the idea, and for evaluating the proposed method a real dataset is used. The data covers an RMC for a period of a month. The results show that proposed method is able to find feasible solution for more than 99% of customers.

ACKNOWLEDGMENT

We acknowledge computing time at the Leonardi cluster in the Faculty of Engineering, the University of New South Wales (UNSW).

REFERENCES

[1] L. Asbach, U. Dorndorf, and E. Pesch, "Analysis, modeling and solution of the concrete delivery problem," *European Journal of Operational Research*, vol. 193, no. 3, pp. 820–835, 2009.

[2] D. Naso, M. Surico, B. Turchiano, and U. Kaymak, "Genetic algorithms for supply-chain scheduling: A case study in the distribution of ready-mixed concrete," *European Journal of Operational Research*, vol. 177, no. 3, pp. 2069–2099, 2007.

[3] V. Schmid, "Trucks in movement: hybridization of exact approaches and

TABLE I
SUMMARY OF RESULTS

		Robust-GA				Random-Solve			
Instance ID	ND	NG	NUAD	ET	Cost	NG	NUAD	ET	Cost
1	19	1085	0	146	224.1	100,000	12	19197	387
2	24	145	0	88	247.6	100,000	16	19922	540
3	25	257	0	65	298.6	100,000	16	19292	453
4	43	589	0	2371	551.2	100,000	33	21700	861
5	64	696	0	354	875.5	100,000	53	25473	1419
6	66	389	0	351	831.8	100,000	54	26493	1372
7	88	1295	0	696	1132.4	100,000	75	28921	2142
8	97	988	0	909	1020.5	100,000	83	30434	2593
9	100	1220	0	16024	886.4	100,000	86	30570	2219
10	101	951	0	14460	1099.4	100,000	87	30830	2591
11	106	6745	0	1983	1435.7	100,000	91	31288	2262
12	113	1568	0	3707	1028.9	100,000	98	31095	2632
13	118	3836	0	19958	951.7	100,000	101	32957	2543
14	119	4646	0	10537	1156	100,000	104	33776	2965
15	126	58432	0	21898	1050.1	100,000	109	38364	2834
16	126	2094	0	18123	11930	100,000	110	35022	3111
17	127	2743	0	23443	1323.8	100,000	112	35647	2953
18	128	2435	0	2566	1269	100,000	112	37068	2816
19	131	8436	0	24618	1160.7	100,000	115	36629	3116
20	134	23231	0	32230	1365.3	100,000	118	38352	3062
21	136	13854	0	27752	1531.2	100,000	119	38171	2882
22	144	14292	0	39373	1374	100,000	127	36079	3231
23	147	2383	0	2256	1608.2	100,000	130	38928	3423
24	153	4382	0	16845	1598.6	100,000	135	40877	3557
25	154	2435	0	25600	1269	100,000	136	41216	3572
26	184	92000	4	89230	2041.4	100,000	172	54462	4086
27	198	68714	0	68678	2030	100,000	178	46809	4389
Average	110	11846	~ 0	17195	1529.3	100,000	96	33317	2518.9

ND = Number of Deliveries per Instance
NG = Number of Generations
NUAD = Number of Unassigned Deliveries
ET = Elapsed Time (Sec)
Cost = Travelled Distance by Trucks (Km)

variable neighborhood search for the delivery of ready-mixed concrete," *Unpublished doctoral dissertation, Universitat Wien, Austria*, 2007.

[4] J. K. Lenstra and A. Kan, "Complexity of vehicle routing and scheduling problems," *Networks*, vol. 11, no. 2, pp. 221–227, 1981.

[5] G. Laporte, "The vehicle routing problem: An overview of exact and approximate algorithms," *European Journal of Operational Research*, vol. 59, no. 3, pp. 345–358, 1992.

[6] G. J. Woeginger, *Exact algorithms for NP-hard problems: A survey*. Springer, 2003, pp. 185–207.

[7] M. Desrochers, J. Desrosiers, and M. Solomon, "A new optimization algorithm for the vehicle routing problem with time windows," *Operations research*, vol. 40, no. 2, pp. 342–354, 1992.

[8] S. Yan, W. Lai, and M. Chen, "Production scheduling and truck dispatching of ready mixed concrete," *Transportation Research Part E: Logistics and Transportation Review*, vol. 44, no. 1, pp. 164–179, 2008.

[9] J. Garcia, S. Lozano, K. Smith, T. Kwok, and G. Villa, "Coordinated scheduling of production and delivery from multiple plants and with time windows using genetic algorithms," in *Neural Information Processing, 2002. ICONIP'02. Proceedings of the 9th International Conference on*, vol. 3. IEEE, Conference Proceedings, pp. 1153–1158.

[10] C.-W. Feng, T.-M. Cheng, and H.-T. Wu, "Optimizing the schedule of dispatching rmc trucks through genetic algorithms," *Automation in Construction*, vol. 13, no. 3, pp. 327–340, 2004.

[11] M. Lu, "Hkconsim: a simulation platform for planning and optimizing concrete plant operations in hong kong," in *Proceedings of International Conference on Innovation and Sustainable Development of Civil Engineering in the 21st Century*, 2002, pp. 278–283.

[12] M. Cao, M. Lu, and J.-p. Zhang, "Concrete plant operations optimization using combined simulation and genetic algorithms," in *Machine Learning and Cybernetics, 2004. Proceedings of 2004 International Conference on*, vol. 7. IEEE, Conference Proceedings, pp. 4204–4209.

[13] M. Lu and H.-C. Lam, "Optimized concrete delivery scheduling using combined simulation and genetic algorithms," in *Proceedings of the 37th conference on Winter simulation*. Winter Simulation Conference, 2005, pp. 2572–2580.

[14] L. Ming and L. Hoi-Ching, "Simulation-optimization integrated approach to planning ready mixed concrete production and delivery: Validation and applications," in *Simulation Conference (WSC), Proceedings of the 2009 Winter*, Conference Proceedings, pp. 2593–2604.

[15] D.-P. Wu, M. Lu, and J.-P. Zhang, "Efficient optimization procedures for stochastic simulation systems," in *Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on*, vol. 5. IEEE,

Conference Proceedings, pp. 2895–2900.
[16] M. Lu, D.-p. Wu, and J.-p. Zhang, "A particle swarm optimization-based approach to tackling simulation optimization of stochastic, large-scale and complex systems," in *Advances in Machine Learning and Cybernetics*. Springer, 2006, pp. 528–537.
[17] M. Lu, F. Dai, and W. Chen, "Real-time decision support for planning concrete plant operations enabled by integrating vehicle tracking technology, simulation, and optimization algorithms," *Canadian Journal of Civil Engineering*, vol. 34, no. 8, pp. 912–922, 2007.
[18] C.-W. Feng and H.-T. Wu, "Integrating fmga and cyclone to optimize the schedule of dispatching rmc trucks," *Automation in Construction*, vol. 15, no. 2, pp. 186–199, 2006.
[19] M. Y. Cheng, H. C. Tsai, and C. L. Liu, "Artificial intelligence approaches to achieve strategic control over project cash flows," *Automation in Construction*, vol. 18, no. 4, pp. 386–393.
[20] C. A. Silva, J. M. Faria, P. Abrantes, J. M. C. Sousa, M. Surico, and D. Naso, "Concrete delivery using a combination of ga and aco," in *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC '05. 44th IEEE Conference on*, Conference Proceedings, pp. 7633–7638.
[21] L. Pan, W. Liya, D. Xihai, and G. Xiang, "Scheduling of dispatching ready mixed concrete trucks trough discrete particle swarm optimization," in *Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on*, Conference Proceedings, pp. 4086–4090.
[22] S. Srichandum and T. Rujirayanyong, "Production scheduling for dispatching ready mixed concrete trucks using bee colony optimization," *American Journal of Engineering and Applied Sciences*, vol. 3, no. 1, pp. 7–14, 2010.
[23] S. Yan and W. Lai, "An optimal scheduling model for ready mixed concrete supply with overtime considerations," *Automation in Construction*, vol. 16, no. 6, pp. 734–744, 2007.
[24] J. H. Holland, *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence*. U Michigan Press, 1975.
[25] R. Poli and W. B. Langdon, "Schema theory for genetic programming with one-point crossover and point mutation," *Evolutionary Computation*, vol. 6, no. 3, pp. 231–252, 1998.
[26] M. Srinivas and L. M. Patnaik, "Adaptive probabilities of crossover and mutation in genetic algorithms," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 24, no. 4, pp. 656–667, 1994.