Just-In-Time Production and Delivery in Supply Chains: a Hybrid Evolutionary Approach*

D. Naso, M. Surico and B. Turchiano
Dipartimento di Elettrotecnica ed Elettronica
Politecnico di Bari, Italy
{naso,turchiano}@poliba.it

U. Kaymak
Erasmus School of Economics
Rotterdam, The Netherlands
u.kaymak@ieee.org

Abstract – *The timely production and distribution of rapidly perishable goods is one of the most challenging logistic problems in the context of supply chain operation. The problem involves several tightly interrelated planning, scheduling and routing problems, each with large combinatorial complexity. From a more practical perspective, the problem calls for a trade-off between risks and returns. To effectively deal with these considerable difficulties, we propose a novel meta-heuristic approach based on a hybrid evolutionary algorithm combined with constructive heuristics for addressing just-in-time production and delivery with time constraints on both the earliness and the lateness of supply. Distribution of ready-made concrete is used as a practical example. A case study based on industrial data illustrates the potential of the proposed approach.*

Keywords: Supply chain management, genetic algorithms, meta-heuristics, concrete delivery.

1 Introduction

Recently, production industry is experiencing a strategic evolution toward the decentralization of many production activities, increasing the importance of supply chains. Supply chains can be viewed as dynamic networks of partially independent production centers that agree to collaborate for pursuing both individual and collective aims. For instance, independent companies that are able to provide complementary services for the production of a given good may take a significant advantage by synchronizing their activities to reduce product lead times or costs. The control and optimization of material, information and financial flows in supply chains currently constitutes an important research field [6].

From the logistic viewpoint, the management of supply chains involves a set of complex and interdependent combinatorial problems such as the acquisition of raw materials, scheduling of production facilities and routing of transport vehicles. Even when considered as independent from the other ones, each of the mentioned logistic problems suffers from a nearly prohibitive combinatorial complexity. However, there is also a strong need for approaches that are capable of finding satisfactory solutions to these complex problems in short computation times. A class of modern meta-heuristic approaches that seems to be particularly suited for dealing effectively and efficiently with the complexity in supply chains is the Genetic Algorithms (GAs). GAs are heuristic search techniques inspired from the principles of survival-of-the-fittest in natural evolution and genetics. They have been used extensively to solve combinatorial problems that cannot be handled by exhaustive or exact methods due to their prohibitive complexity. When properly configured, GAs are efficient and robust optimization tools, because they do not explicitly require additional information (such as convexity, or availability of derivative information) about the objective function to be optimized. However, GAs are generally slow, they require large numbers of iterations, and suffer from specific problems that may cause premature convergence in suboptimal solutions. Therefore, the average time that a well-configured GA would need to search for a satisfactory solution of the entire supply-chain problem (with its many decision variables) is too high for practical use in a real industrial context, where decision-algorithm must provide a solution in relatively short times. For this reason, we propose a novel meta-heuristic approach based on a hybrid evolutionary algorithm combined with constructive heuristics for addressing planning, scheduling and routing for just-in-time production and delivery. In this approach, we use a GA to perform part of the optimization, while the remaining part of the scheduling problem is handled by constructive heuristic algorithms. This approach leads to a hybrid evolutionary algorithm in which the GA constitutes the core of the search strategy, while multiple heuristic rules called in specific circumstances contribute to reconstruct a feasible solution that satisfies all the constraints and objectives of the problem. In this respect, the proposed approach is significantly different from other recent applications of GAs and other meta-heuristic approaches to complex combinatorial problems.

In this paper, distribution of ready-made concrete is used as a practical example to illustrate the potential of the proposed approach. This is a problem with time constraints on both the earliness and the lateness of the supply. Note, however, that even though this paper is mainly devoted to the problem of ready-mixed concrete supply, both the proposed

*0-7803-8566-7/04/$20.00 © 2004 IEEE.

model and the resolving strategy are fairly general, and can be easily extended to address a variety of analogous just-in-time distributed production and delivery problems with or without time constraints on both the earliness and the lateness of supply.

The outline of the paper is as follows. Section 2 describes the concrete delivery problem that is used as a case in this paper, while Section 3 describes the mathematical model for the scheduling problem in the supply chain. The proposed approach to address this problem is described in Section 4. Experimental results from a case study based on real-world data can be found in Section 5. Finally, conclusions are given in Section 6.

2 Concrete delivery

Producing and distributing ready-made concrete is a complex logistic problem that involves several interdependent assignment and scheduling problems. Moreover, the specific characteristics of the produced material and its utilization in construction entail a large number of additional technical constraints that must be taken into account. Typically, concrete is prepared at a production center, mixed with water and then transported directly to the customer for immediate use. Concrete is a perishable material, and once water is added to the dry mix of materials, it retains its properties only for a couple of hours. An overview of the main characteristics of cement production and delivery can be found in [5].

We consider a network of D suppliers or Production Centers (PCs) located in a given geographical area. Each PC is equipped with a single loading dock, where concrete is loaded on the trucks for delivery. We assume that the producer knows the customers' orders in advance on the delivery day. Each demand consists minimally of a delivery moment (date and time), type of concrete, required amount and delivery location. Based on this data the producer has to (1) assign the demands to the PCs and (2) deliver the concrete batches by using the truck fleet.

There are a number of constraints regarding the trucks that must be taken in account when building both the production and the delivery schedule.

1. A truck cannot transport more concrete than its capacity. Thus orders that exceed the maximal truck capacity C_{max} must be divided into multiple sub-deliveries (jobs).

2. A truck can service only one order at a time. It is not possible to service multiple small orders by the same truck during the same delivery. Hence, a small order often implies that a truck will be loaded only partially during the delivery.

3. Each truck starts the day from a base location and must return to the same base location at the end of the day.

4. Each truck has a specified maximal unloading rate.

There are also a number of constraints regarding the loading and unloading of the vehicles.

1. The loading of a delivery takes place in a single loading dock.

2. A loading dock can service only one truck at a time.

3. At the delivery location, only one truck can be unloaded. If two trucks arrive at the location, one must wait until the other one is unloaded.

4. The vehicle can leave the delivery location only after it has been unloaded completely. Partial unloading is not possible.

5. The rate of unloading cannot be faster than the capacity specified by the client.

6. If an order is so large that it requires multiple deliveries, the first delivery must arrive at the agreed time.

7. Subsequent deliveries of a multi-delivery order must start immediately after the previous delivery is unloaded.

For each delivery, the truck must first be loaded, driven to the delivery location, possibly wait a few minutes for other activities to complete, unload and drive away. If the order requires multiple deliveries, a loaded truck must be available at the site when the preceding truck has ended its unloading, in order to guarantee the continuity of the unloading. The optimization of the delivery requires a schedule characterized by (1) minimal costs for transportation, and (2) little waiting time for the customers. On-time delivery is essential. If a truck arrives early, the concrete placement crew may not yet be ready or the preceding operations may have not been completed, thus keeping the vehicle idle. If a truck arrives late, the continuity of unloading is violated, and if the delay exceeds the concrete setting time the entire load has to be disposed.

In practice, there are also a number of other restrictions that should be taken into account when generating the final schedule.

1. The full capacity of a truck cannot be used with certain types of concrete. In this case, the truck can only be loaded up to a certain fraction.

2. The client may require a truck to arrive extra early for logistic reasons at the construction site.

3. The total loading time for a truck consists of a fixed part (independent of the loading rate) and a flexible part that depends on the required volume and the loading rate of the loading dock.

4. A vehicle can be used for a certain number of hours a day. If there are delays or a vehicle is needed longer, then an overtime must be paid.

5. The customer can specify the maximum amount per delivery. In that case, each delivery of the order should not exceed this amount.

6. An order is sometimes assigned to a specific production center. In that case, only that center prepares the concrete for that order. It is also possible that certain production centers are excluded from servicing a specific order.

Note that all these constraints can be taken into account when generating the final schedule in our approach. However, for the sake of brevity, we describe, in the next section, only a simplified mathematical model of the problem.

3 Simplified mathematical model

Assume there are R different demands to be scheduled. Each demand r is characterized by the location of the customer, the maximal unloading rate UR_r, the maximal delivery size Mds_r, the quantity of concrete Q_r, the setting time $Tset_r$ of the concrete, the percentage Per_r of the truck that must be left empty, the fixed waiting time Fix_r at the customer, and the earliest and latest delivery times (EDT_r and LDT_r) for accomplishing the delivery. We assume a homogeneous fleet of vehicles (same maximum capacity C_{max} and average speed V). If a demand exceeds the capacity of a single truck, it is equally divided in Z_r jobs, each of which will be delivered by a single vehicle.

$$Z_r = \left\lceil \frac{Q_r}{\min\{C_{max}(1-Per_r), Mds_r\}} \right\rceil . \tag{1}$$

Each job is identified by a sequential number and its related index is i. The first job of a demand r is f_r and the last one is l_r. The total number of jobs is N.

$$i \in \left\{ \underbrace{1,\dots,Z_1}_{r=1}, \underbrace{Z_1+1,\dots,Z_1+Z_2}_{r=2}, \cdots, \underbrace{\sum_{r=1}^{R-1} Z_r + 1,\dots,N}_{r=R} \right\}$$

If the D PCs are not able to supply all the demanded concrete, a part of the requests (and their delivery) will be outsourced to external companies, with a consequent loss of income. If the fleet of K trucks is not able to deal with all the deliveries, one can hire new vehicles at an additional cost. We define $K = K_c + K_o$, where K_c is the number of trucks owned by the company (the internal fleet) and K_o is the number of additionally hired trucks (usually $K \ll N$). When a job is assigned to a truck it is referred to as a task with related index m. Then, our model considers the following decision variables.

$Y_{id} \in \{0,1\}$	if job i is produced at the depot d, $Y_{id} = 1$ and $i \in \Gamma_d \subseteq \{1,\dots,R\}$; otherwise $Y_{id} = 0$.
$Y_{oi} \in \{0,1\}$	if the production of job i is outsourced, $Y_{oi} = 1$; otherwise $Y_{oi} = 0$.
$X_{ikm} \in \{0,1\}$	if job i is assigned to truck k as m-th task, $X_{ikm} = 1$; otherwise $X_{ikm} = 0$.

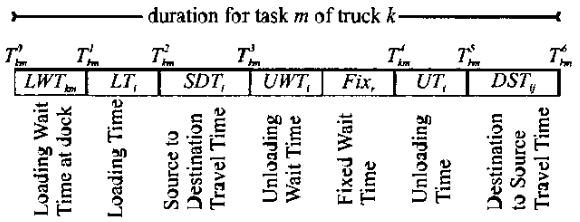

Figure 1: Sequence of operations for a single truck.

Each task can be divided as shown in Fig. 1. Further, let LR_d be the concrete mixing/loading rate of the depot d, and let FLT_d be the fixed component of the loading time at the same depot. The total loading time LT_i for job i is then

$$LT_i = \sum_{d=1}^{D} Y_{id} \cdot \left(FLT_d + \frac{Q_r}{Z_r} \frac{1}{LR_d} \right), \tag{2}$$

while its unloading time is

$$UT_i = \frac{Q_r}{Z_r} \frac{1}{UR_r}. \tag{3}$$

Since each depot can service only one truck at a time, we must ensure that the loading windows do not overlap. These windows will last from SLT_i (starting of the loading time) to ELT_i (end of the loading time). For each job we must also guarantee that the SLT_i is not too early, because hydration must not start before the full concrete delivery. Hence,

$$SLT_i \geq LDT_{r(i)} - (l_{r(i)} - i)\, UT_i - Tset_{r(i)}. \tag{4}$$

Similarly, ELT_i can not be so late that the succeeding jobs of the same demand can not be unloaded within the customer's specified time window. Thus,

$$ELT_i \leq LDT_{r(i)} - SDT_i - Fix_{r(i)} - \left(l_{r(i)} - i + 1\right) UT_i. \tag{5}$$

To accomplish demands requiring multiple deliveries preserving the continuity of the unloading phase, we must guarantee that, for two successive jobs i and $i+1$

$$\sum_{k=1}^{K} \sum_{m=1}^{M} X_{ikm} T^5_{km} = \sum_{k=1}^{K} \sum_{m=1}^{M} X_{(i+1)km} T^4_{km}. \tag{6}$$

We model the integrated supply chain scheduling as a cost minimization problem. The cost function is composed of three terms,

$$C_{Total} = C_{transport} + C_{idle} + C_{extra}. \tag{7}$$

The first term takes into account the transportation costs. From a global viewpoint, the production of the entire supply chain should be organized so as to minimize the delivery costs. The second term takes into account the loading and unloading waiting times. Waiting times should be minimized because they typically represent a loss (the more the

Customer's Request-to-Depot Assignment						Priority of request in schedule construction					
r_1	r_2	r_5	r_4	r_5	r_5	p_1	p_2	p_3	p_4	p_5	p_6
1	3	3	1	2	2	4	5	6	1	5	2

Figure 2: A generic chromosome.

waiting times, the lower the resource utilization). The third term accounts for the outsourced jobs and the hired trucks (other than the truck drivers overtime). Solutions requiring a different number of outsourced jobs or hired trucks can be found for any given set of demands. The more the schedule is optimized, the lower the amount of requested outsourcing.

4 Hybrid evolutionary approach

The task of finding a feasible schedule for the overall supply chain is a problem of prohibitive complexity. To deal with this complexity, we propose in this section a hybrid evolutionary approach combined with constructive heuristics. A GA is used to optimize the assignment of demands to depots (Y_{id}) and the order of priority by which the demands are scheduled for production. Then, a constructive heuristic algorithm is used (1) to build schedules satisfying all the described constraints and (2) to assign the jobs (that are not outsourced) to the trucks.

4.1 Genetic Algorithm

Every GA requires the encoding of the candidate solution in a string of symbols. The chromosomes encode part of the decision variables and some variables that provide input to the constructive heuristic algorithm, which determines the actual schedule and it is run each time the fitness of a chromosome is computed. In this way, the search space for the genetic algorithm is kept small, while all constraints for the scheduling problem are satisfied.

Each chromosome is made up of two separate parts, both containing R elements, as described in Fig. 2 for $R = 6$. The first part defines the assignment of demands to the depots. The l-th gene of this part indicates the PC to which request r_l is assigned. The second part of the chromosome establishes the order in which the R requests are considered in the construction of the complete schedule of the production chain. In other words, the first part of the chromosome defines initial values for the decision variable Y_{id}, while the values of all decision variables including Y_{oi} and X_{ikm} are fixed by the constructive heuristic procedure. Here follows the specific characteristics of our GA.

1. The initial population *Pop* is made up of 100 individuals and it is created randomly.

2. The fitness of each new individual in *Pop* is evaluated by our constructive heuristic that builds the relative schedule and returns the value of the function cost.

3. Tournament selection [3] is used. It selects randomly pairs of individuals and the individual with the higher fitness is passed on to the next generation.

4. A new crossover operator is designed taking into account the specific structure of the chromosomes. It selects randomly a crossover point. If this point falls in the first half of the chromosome, it performs a standard single-point crossover on the first part of the chromosome (request to depot assignment), otherwise it performs an order-based crossover on the remaining part [4].

5. Similarly to the crossover, we selectively apply two different mutation operators. A gene in the chromosome is selected randomly. If it belongs to the first part, the gene is replaced by a randomly extracted integer between 1 and D. Otherwise, the inversion mutation (two randomly selected genes are swapped in the sequence) is applied to the order-based part [1].

6. In all cases, we stopped our algorithms after 200 generations.

4.2 Constructive heuristics

A schedule construction algorithm (SCA) is used to schedule the production and the delivery of all the jobs. The first part of the algorithm (depot SCA or DSCA) processes the demands following the order of priority specified by the chromosome. For each demand, the DSCA checks the distance between the PC assigned in the chromosome and the customers, redirecting the demand to the nearest depot if such distance does not allow full unloading before T_{set}. The algorithm computes the SLT of the first job of the demand so that its unloading starts exactly at the customer-specified EDT. If an overlap with a previously assigned job occurs, the DSCA tries to shift forward the SLT until the conflict is solved. Of course the relative unloading phase is shifted forward as well. If the specified LDT is violated as a result of shifting, the DSCA makes another try, this time shifting backward the loading window. However the truck now arrives at the delivery location before the optimal time and, consequently, it introduces an idle waiting time. Among other things, we must guarantee that the latter is not so great that the time between the SLT and the end of the unloading becomes greater than T_{set}. If this occurs the DSCA reassigns the demand to another PC, because we assume that the previously scheduled jobs (the higher priority ones) cannot be modified to accommodate successive demands (lower priority ones). Note however that the priority order is not fixed and it is optimized by the GA. If no adjustment guarantees a successful schedule of the job, the DSCA attempts to assign the second job of the demand to the PC, verifying at the same time that the preceding jobs can be scheduled within the described constraints. If also the second job cannot be assigned to the depot, the algorithm tries with the subsequent ones, until either one of the jobs is assigned to the depot, or none of the jobs composing a demand can be scheduled on the PC. In the latter case (100% jobs to be reassigned elsewhere), the gene of the chromosome is modified and the procedure starts investigating the succeeding PCs in the order of shortest distance from the customer's site. After successfully assigning at least one of the jobs of the demand, the DSCA tries to assign all the remaining jobs to the same PC, with the additional constraint that the unloading of each job must start when the preceding one ends.

In a second phase, the DSCA tries to place the jobs unscheduled for production so far on other PCs, considering them in the order of increasing distance from the customer's site. If at the end of this phase there are still jobs not assigned to any PC, the DSCA tries to "force" their insertion at a PC (starting from the nearest) at the time that guarantees zero unloading waiting time for the trucks. Only during this last phase the DSCA is allowed to shift-backward the jobs previously scheduled. If the insertion does not cause a violation of the T_{set} by any of the shifted jobs and it is less expensive than outsourcing the job the schedule is modified accordingly. Otherwise the DSCA tries with the next PC. If a job cannot be successfully reassigned to one of the available depots of the company at the end of this phase, then it is set for outsourcing. Then, the second part of the SCA (truck SCA or TSCA) schedules the deliveries so that, for each job, a truck is available at the supplying PC no later than the scheduled SLT. In a first phase, the TSCA tries to assign the jobs produced at a given PC to the fleet of vehicles whose base location is the same PC.

To illustrate the allocation procedure, let us firstly define the set of hypothetically available trucks at a depot at the generic time t. This set is composed of the trucks that either have not left their base PC from the beginning of the working day or have already completed some transport operations and can return to the PC before time t. When both types of trucks are available, the TSCA always tries to assign the jobs to the previously used vehicles first, in order to actually use the minimal amount of trucks for servicing all the requests. At the beginning of the working day all the trucks of the PC are idle. The TSCA allocates the first jobs produced in the working day to the idle trucks until some truck returns after completing one task. From that time on, the TSCA gives higher priority to trucks with the latest return time. This assignment strategy is referred to as *Shortest* (truck) *Idle Time* (SIT), because the truck with the smallest idle time at the PC is the one assigned first. The reason of using this strategy is twofold. Firstly this strategy helps to use a minimal number of trucks. Secondly, instead of evenly distributing the idle times among trucks, the SIT strategy causes some trucks to have longer idle times between assigned services. This is useful in the succeeding phase, because other jobs can fit during these long idle times. When a job is assigned to a truck, the variable X_{ikm} is updated accordingly. If the TSCA is unable to find a truck for a given job, the job is temporarily marked as undeliverable, and its assignment is postponed. The first part of the TSCA proceeds with the job-to-truck assignment until it has inspected all the jobs. If at the end of this phase the set of unassigned jobs is not empty (usually for the unavailability of trucks at some PCs) the TSCA will assign them during a second phase. All the unassigned jobs are considered in the order of increasing SLT. The TSCA considers the set of remaining trucks sorted by completion time of the last operation and tries to assign the current job to the first truck in this list inspecting various insertion possibilities (either placing it after the last job, or inserting it between two previously assigned jobs). If no insertion meets all the constraints, the procedure considers the next truck in the list. If the job cannot be assigned to any truck in the list, a new one is hired, so that, at the end of this phase, all the jobs that have not been outsourced are assigned for delivery.

Table 1: Cost parameters.

CP	10	cost/Km of travel of trucks
CP	15	penalty for idle time
PT	2000	loss of income for m^3 of outsourced concrete
HC	10000	cost of an hired truck
XTR	5	cost (extra pay) for each minute of working out of the standard working time

5 Case study

As a case study, we consider a supply chain composed of five PCs located in the Netherlands. The fleet of trucks consists of 49 vehicles housed in two PCs. The customers are spread over the area surrounding the PCs of the supply chain. As discussed in the literature [2], there is a high density of demand between 7:00 and 9:00 hours, and between 13:00 and 15:00 hours. The objective of scheduling is to minimize the costs associated with the delivery. Table 1 summarizes the values of the main cost parameters in normalized cost units (CU).

The trucks have a maximal capacity C_{max} of 10 m^3 and an average speed V of 60 Km/h. In general, customer requests have very narrow time windows, which impose to schedule the delivery of the first job very close to the EDT. The concrete setting time T_{set} used in our model is 150 minutes. The working day for a truck is between 5:00 AM to 4:00 PM, and if some truck is scheduled to work outside this window, an additional cost must be paid.

To test the effectiveness of the proposed hybrid GA approach, we have compared it with four different scheduling policies obtained by applying assignment criteria suggested by experts. All of the considered alternative policies give higher priority to larger orders, since this criterion is the one used by most companies. Demand assignments take into account the distances from the PCs to the customers' sites or the actual workload of the PC, while the trucks are assigned based on their idle times. The four policies used here for comparison are the following.

1. *SD/SIT (Shortest Distance/Shortest Idle Time).* This heuristic criterion tries to allocate each job to one of the

Table 2: Summary of scheduling results obtained on a reference case (all times in minutes)

variant	SD/SIT	SD/LIT	SW/SIT	SW/LIT	GA
Jobs outsourced	3 (28 m^3)	3 (28 m^3)	6 (48 m^3)	3 (28 m^3)	0 (0 m^3)
Hired trucks	17	22	21	17	15
Overtime (t)	0	0	21	22	55
Empty trips (t)	5290	5119	7123	7242	5479
Loaded trips (t)	3129	3129	6716	6719	5068
Wait time (t)	7214	15315	4803	10657	4665
Total cost (CU)	417952	587757	515856	624856	325720

PCs, considering them in the order of increasing distance from delivery location. The satisfaction of all the constraints is delegated to a schedule builder. If no PC can supply the job, it is outsourced. Once all the jobs are scheduled, they are assigned to trucks, giving higher priority to the ones that have been idle for the shortest time.

2. *SD/LIT (Shortest Distance/Longest Idle Time).* This is a variant of the previously described policy in which only the truck assignment is changed by trying to load each job on the truck that has been idle for the longest time.

3. *SW/SIT (Smallest Workload/ Shortest Idle Time).* This heuristic algorithm tries to assign each job to PCs considering them in the order of increasing workload already assigned. The truck assignment strategy is the same as in the case 1.

4. *SW/LIT (Smallest Workload/ Longest Idle Time).* This is a variant of policy 3 in which only truck assignment strategy is changed to LIT.

In order to provide a clear idea of the results obtained by the hybrid GA approach, let us focus on the scheduling of the production of a demand pattern observed during a typical working day of the supply chain. Table 2 summarizes the results obtained by the five policies considered. This case considers 71 demands for a total amount of 2116.3 m^3 of ready-made concrete, divided in 258 jobs. It is worth noting that the GA-based policy is able to find a schedule that does not entail outsourced jobs, while also minimizing the number of hired trucks necessary to deliver the concrete to the customers. The total cost of the solution obtained by the GA is about 20% lower than the one provided by SD/SIT, which is the most effective heuristic that we have compared with the GA. In particular, being focused on the optimization of truck routes, the SD/SIT is able to provide the smallest cost associated to transportation, at the expense of longer overall amount of waiting times. It should be remarked that the large amount of waiting times is not evenly distributed between the operations, so the solution found by SD/SIT is not significantly more delay-tolerant than the one obtained with the GA. On the contrary, the job distribution obtained with the GA provides a considerably increased overall length of truck routes, which is fully compensated by the ability to assign all the production requests to the 5 PCs of the supply chain.

6 Conclusions

We have considered a novel meta-heuristic approach based on a hybrid evolutionary algorithm combined with constructive heuristics for addressing just-in-time production and delivery with time constraints on both the earliness and the lateness of supply. Distribution of ready-made concrete is used as a case study. It is shown that such a hybrid approach is able to provide an effective scheduling algorithm based on a realistic model of the supply chain. The scheduling algorithm combines a GA and a set of constructive heuristics, which guarantee the determination of a feasible schedule for any given set of orders. A comparison of the proposed method with four different heuristic scheduling algorithms have shown its superior performance. The superior performance manifests itself on two counts. Firstly, the amount of requests that are redirected to external companies, or that need additional hired trucks for their delivery, is in general very small compared to the other scheduling strategies. Secondly, the proposed model allows the definition of safety margins for minimizing the effects of transportation delays. In the future, we will investigate the robustness of the solution determined by the proposed approach to disturbances in the parameters, such as the average truck speed.

References

[1] H. Ishibuschi, T. Yoshida, and T. Murata. Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling. *IEEE Transactions on Evolutionary Computation*, 7(2):204–223, 2003.

[2] N. F. Matsatsinis. Towards a decision support system for the ready concrete distribution system: a case of a greek company. *European Journal of Operational Research*, 152(2):487–499, Jan. 2004.

[3] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer, Berlin, third (extended) edition, 1996.

[4] A. C. Nearchou. The effect of various operators on the genetic search for large scheduling problems. *International Journal of Production Economics*, 88(2):191–203, 2004.

[5] I. D. Tommelein and A. Li. Just-in-time concrete delivery: mapping alternatives for vertical supply chain integration. In *Proceedings of the Seventh Annual Conference of the International Group for Lean Construction IGLC-7*, pages 97–108, University of California, Berkeley, California, July 1999.

[6] N. Viswanadham. The past, present, and future of supply–chain automation. *IEEE Robotics & Automation Magazine*, 9(2):48–56, June 2002.