GENETIC ALGORITHMS IN SUPPLY CHAIN SCHEDULING OF READY-MIXED CONCRETE

D. Naso, M. Surico, B. Turchiano and U. Kaymak

ERIM REPORT SERIES *RESEARCH IN MANAGEMENT*	
ERIM Report Series reference number	ERS-2004-096-LIS
Publication	October 2004
Number of pages	47
Email address corresponding author	kaymak@few.eur.nl
Address	Erasmus Research Institute of Management (ERIM) Rotterdam School of Management / Rotterdam School of Economics Erasmus Universiteit Rotterdam P.O. Box 1738 3000 DR Rotterdam, The Netherlands Phone: +31 10 408 1182 Fax: +31 10 408 9640 Email: info@erim.eur.nl Internet: www.erim.eur.nl

Bibliographic data and classifications of all the ERIM reports are also available on the ERIM website:
www.erim.eur.nl

ERASMUS RESEARCH INSTITUTE OF MANAGEMENT

REPORT SERIES
RESEARCH IN MANAGEMENT

BIBLIOGRAPHIC DATA AND CLASSIFICATIONS		
Abstract	The coordination of just-in-time production and transportation in a network of partially independent facilities to guarantee timely delivery to distributed customers is one of the most challenging aspects of supply chain management. From the theoretical perspective, the timely production/distribution can be viewed as a hybrid combination of planning, scheduling and routing problem, each notoriously affected by nearly prohibitive combinatorial complexity. From a practical viewpoint, the problem calls for a trade-off between risks and profits. This paper focuses on the ready-made concrete delivery: in addition to the mentioned complexity, strict time-constraints forbid both earliness and lateness of the supply. After developing a detailed model of the considered problem, we propose a novel meta-heuristic approach based on a hybrid genetic algorithm combined with constructive heuristics. A detailed case study derived from industrial data is used to illustrate the potential of the proposed approach.	
Library of Congress Classification (LCC) LCC Webpage	Mission: HF 5001-6182	
	Programme: TX301-329	
	Paper:	HB 143.7 Optimization techniques
Journal of Economic Literature (JEL) JEL Webpage	Mission: M	
	Programme : M 11, R 4	
	Paper:	C 61 Optimization, Programming models
Gemeenschappelijke Onderwerpsontsluiting (GOO)		
Classification GOO	Mission: 85.00	
	Programme: 85.34	
	Paper:	85.03 Operations research
Keywords GOO	Mission: Bedrijfskunde / Bedrijfseconomie	
	Programme: logistiek, management	
	Paper: Ketenbeheer, just-in-time-productie, modellen, wiskundige programmering	
Free keywords	Supply chain management, genetic algorithms, meta-heuristics, concrete delivery	

Genetic Algorithms in Supply Chain Scheduling of Ready-Mixed Concrete

D. Naso [a] M. Surico [a] B. Turchiano [a] U. Kaymak [b]

[a] Politecnico di Bari, Dipartimento di Elettrotecnica ed Elettronica
Via Re David, 200 70125 Bari, Italy
Email: naso@poliba.it

[b] Department of Computer Science, Erasmus School of Economics
P.O. Box 1738, 3000 DR Rotterdam, The Netherlands
Email: u.kaymak@ieee.org

Abstract

The coordination of just-in-time production and transportation in a network of partially independent facilities to guarantee timely delivery to distributed customers is one of the most challenging aspect of supply chain management. From the theoretical perspective, the timely production/distribution can be viewed as a hybrid combination of planning, scheduling and routing problem, each notoriously affected by nearly prohibitive combinatorial complexity. From a practical viewpoint, the problem calls for a trade-off between risks and profits. This paper focuses on the ready-made concrete delivery: in addition to the mentioned complexity, strict time-constraints forbid both earliness and lateness of the supply. After developing a detailed model of the considered problem, we propose a novel meta-heuristic approach based on a hybrid genetic algorithm combined with constructive heuristics. A detailed case study derived from industrial data is used to illustrate 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. Due to the various, extremely challenging problems related to collaboration, cooperation, information sharing, synchronization of logistic and production activities, research areas such as supply chains, virtual enterprizes, global manufacturing networks and similar forms of organization have gained a prominent role in the field of industrial automation. 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. Also companies providing the same type of services may find convenient to form strategic consortia and satisfy market requests that exceed the capacity of each individual company. The inter-company and company-to-customer relationships are also experiencing a strong evolution due to the developments in computer and communication technologies, including large scale distributed information networks such as internet. All these aspects have generated a multi-disciplinary research area involving the control and the optimization of material, information and financial flows [17].

From the logistic viewpoint, the management of supply chains involves a set of complex and interdependent combinatorial problems (e.g. acquisition of raw materials, scheduling of production facilities, routing of transport vehicles, etc.). Even when considered as independent from the other ones, each of the mentioned logistic problem suffers from a nearly prohibitive combinatorial complexity. However, there is also a strong need for approaches that are capable of finding satisfactory solutions in short computation times, since these environments are continuously subject to unpredictable dynamic changes (new orders, delays, failures) that may require a sort of continual revision of the planned solution.

In this paper, we focus on a supply chain for the production and distribution of ready-made concrete. The supply chain consists of a network of independent and distributed production centers serving a set of customers distributed across a predefined geographical area surrounding the nodes of the supply chain. Each customer contacts one of the nodes of the supply chain (in general the closest one) to place an order that has to be either accepted or refused. Most production centers in the chain own a fleet of trucks to deliver the produced good to the customer, but a few centers do not own a local fleet for transportation and explicitly rely on the other nodes for the delivery of their production. By joining the supply chain, each production center agrees to convey the received demands to a central planning system, which is in charge of scheduling the production on the various centers in order to optimize the operation of the entire supply chain. This means that after the optimization, the company producing a given demand may no longer be the one that received the order, if this leads to a better overall schedule. The considered problem is made particularly challenging by the fact that the produced good is a perishable material that has to be mixed on-demand and delivered within strict time-windows to customers locations. The individual goal of each production center is to accept and satisfy the maximum number of requests, guaranteeing the timeliness of the deliveries at the minimum overall cost. Some of the received requests may partially exceed the capacity of the contacted production center, but the center may still accept the request and redirect part of the order to another node of the supply chain, or outsource it to an external company. It should be noted that the goal of each production center is composed of inherently conflicting aspects, because on the one hand the maximum utilization of resources implies the reduction of idle and waiting times, and on the other hand these time intervals are the only actual safety margins that make the schedule tolerant to transportation delays or other unexpected circumstances. In any case, since the costs associated to late deliveries can be extremely high, an effective tradeoff between schedule robustness to perturbation and production costs must necessarily be found.

At present, many companies tend to rely on skilled operators that set up an initial planning of production, also estimating which customers request can be accepted, and which has to be rejected [1, 10]. Other companies prefer to plan their operations on short production horizons, sacrificing the optimization on longer horizon to achieve a reduced risk of delayed delivery [15]. However, to authors' knowledge, there is no specific approach in the extremely rich literature on process scheduling and vehicle routing with time-constrained deliveries that can be directly (i.e. without substantial modifications) used to address the considered problem, which suffers from both the typical combinatorial complexity of constrained assignment problems, and from a very large number of specific peculiarities that must be addressed with ad hoc approaches. Thus, the main contribution of this paper is the development of an effective strategy to systematically model and solve the just-in-time production and supply of perishable goods in a efficient, reliable, and systematic way, so as to bridge the gap between industrial practice and technical research. The development of such an effective approach also improves the sustainability of the supply chain management solutions by increasing the utilization of equipment and by decreasing the demand on scarce resources.

This manuscript focuses on the development and validation of an efficient heuristic approach to address the following strongly interrelated problems.

1. Assigning customers' demands of concrete to a number of geographically distributed storage and production centers.

2. Scheduling the available fleet of (company-owned or hired) transport vehicles to deliver all the produced quantities.

To pursue this objective, in the first part of this paper we develop a detailed mathematical model of the considered problem, explaining the available decision variables and the main constraints of the problem. The development of an extensive model constitutes the first important contribution of this paper with respect to the related literature, which is mainly focused on simplified formulations taking into account only a part of the considered problem. Subsequently, the paper focuses on the development of an algorithm belonging to a class of modern problem-solving metaheuristics that seem particularly suited to the considered problem, commonly referred to as Genetic Algorithms (GAs). GAs are heuristic search techniques inspired from the principles of survival-of-the-fittest in natural evolution and genetics, which have been extensively used 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. For this reason, in the last decade, they have been applied to a considerable variety of problems, including scheduling and vehicle routing problems that are partially related to our case.

It should be remarked that GAs are not immune from drawbacks: they are generally slow, they require large numbers of iterations, and suffer from specific problems that may cause premature convergence in suboptimal solutions if not carefully handled with ad hoc methods. Due to the combinatorial nature of our distributed scheduling-routing problem, when considering a real-world daily scenario, the considered problem may become prohibitive also for a GA. In fact, a generic solution of our problem involves

1. assigning demands to production center,
2. scheduling the production at each center,
3. scheduling the loading operations,
4. routing vehicles from (to) production centers to (from) depots,
5. scheduling the unloading operations at each customer's site.

In conclusion, the average time that a well-configured GA would need to search for a satisfactory solution of the entire supply-chain problem may be too high for a practical use in a real industrial context, where decision-algorithm must provide a solution in relatively short times. For this reason, rather than a conventional use of the GA as a global optimizer of all the free decision variables of the problem, in our research the GA is only used to address a part of the whole problem. Namely, we use the GA to perform demand-to-production center assignment, and the production sequencing at each center, while the remaining part of the whole 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. In this respect, the proposed approach is significantly different from other recent applications of GAs and other meta-heuristics approaches to complex combinatorial problems sharing some similarities with the ready-mixed cement supply. To our best knowledge, this manuscript is among the first attempts to address the extremely complex scheduling problem of an entire supply-chain for just-in-time production by explicitly taking into account all the constraints and requirements of the real-world scenario with an hybrid meta-heuristic strategy based on GAs. Finally, it is also important to underline that, even though this paper is mainly devoted to the problem of ready-mixed concrete supply, both the proposed 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.

The paper is organized as follows. Section 2 gives an overview of the related literature, while Section 3 describes the mathematical model of the considered problem. Section 4 introduces the proposed hybrid meta-heuristic approach, illustrating the main GA-based search engine, and the additional schedule construction heuristics in separate subsections. Section 5 describes the case study, summarizing the main results of our approach in comparison with other scheduling heuristics. It also provides a detailed analysis of the tolerance of the optimized schedules to delivery delays. Finally, Section 6 provides the concluding remarks.

2 Literature Overview

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. An overview of the main characteristic of cement production and delivery is provided in [15], which illustrates the reasons that make concrete production possible only on a just-in-time basis, and surveys the various types of vertical supply chain organizations that can be adopted by production companies and customers to pursue their respective objectives. Reference [15] also provides a discussion on how information and material flows can be optimized by placing materials, and time-buffers at strategic locations in construction processes. In this discussion, the authors remark that the most common case is where the batch plant also delivers the mix to the contractor's project site, i.e. the case considered in this paper.

The problem of cement delivery is also considered in [10]. After giving a general discussion of the main peculiarities of the problem, the author focuses on the problem of routing two different types of vehicles: the cement carriers for the delivery of the concrete to customers sites, and the pumps that may be necessary in some sites to unload the cement from the trucks. The routing problems for each type of vehicle have different characteristics, because the carriers must load the cement at a depot, unload it at its destination and then return to the same depot for a new load, while the pumps are allowed to move from one customer to another one. The author proposes a decision-support system to solve both routing problems with a heuristic approach that iteratively improves an initial assignment done by plant managers.

It can be easily noted that the just-in-time supply problem considered here also shares many common aspects with a large number of related logistic problems that have been considered in technical literature. For instance, the assignment of trucks for transportation can be modelled as a vehicle routing problem (VRP), which is among the most thoroughly investigated problems of operations research. The version of the VRP more related to our case study concerns the routing of a fleet of vehicles with time-window constraints and is known as Multi-Depot Multi-Vehicle Routing Problem with Time Windows. The nearly prohibitive combinatorial complexity of this problem makes analytical approaches impracticable, especially for cases of a size comparable to our problem. Thus, available literature focuses on heuristic approaches capable to achieve satisfactory solutions in acceptable times. Comprehensive surveys about exact and heuristic methods to deal with the VRP are available in [5, 9, 16]. A study on vehicle capacity planning system that is considerably related to our problem is provided in [7]. The authors model the problem of container transportation as a vehicle routing problem, also taking into account several time-window constraints. As in our case, each carrier starts from a depot, delivers the unit load (container), and then reaches another depot for a new task. The objective function is to minimize the sum of the cost of delivering the containers by company-owned trucks, and the cost of outsourcing the requests that cannot be satisfied by the company. The authors propose a heuristic approach based on Tabu Search, which is able to determine solutions that are significantly better than those provided by the existing rules adopted by the company.

Several approaches for dealing with similar problems have been proposed in recent literature concerning the scheduling of production centers. As in the case of routing, most of the recent literature overcomes the extreme combinatorial complexity with heuristic solutions. More specifically, there has been an increasing interest towards search algorithms inspired from the principles of survival-of-the-fittest in natural evolution and genetics, generally referred to as Evolutionary Computation [11]. Since these heuristic algorithms are also the main optimization engine of the approach proposed in this paper, we now pay particular attention to some recent references that describe an evolutionary approach for problems related to the supply-chain scheduling considered here.

The allocation of a set of independent jobs with delivery time constraints to a set of distributed plants presents some similarities with scheduling problems with earliness / lateness penalty considered in the context of single or parallel machine scheduling. GAs lend themselves to attack such type of problems as suggested in e.g. [6]. The authors devise a scheduling algorithm composed of a timing algorithm that computes the optimal start time of each job, and a sequencing strategy to determine the processing order based on a GA. The proposed method offers significant improvement with respect to other available heuristics, and provides nearly optimal solutions as confirmed by a comparison with an exact algorithm. The problem of scheduling independent jobs on identical parallel machines to minimize earliness / tardiness with a GA has been considered in [2]. Two GAs, one without crossover, and the other one using a crossover operator devised by the authors for the specific problem are proposed and evaluated against a neighborhood exchange search algorithm. While the latter algorithm provides better results for small instances, the GAs outperform such search algorithm in larger-sized, more difficult problems, providing improvements that increase with the problem size. The scheduling of identical parallel machines with a GA has been considered also in [12], where the objective is formulated as the minimization of the makespan, and the performance of the GA is compared against Simulated Annealing and other available heuristics. Also here, the proposed GA outperforms the other methods suggested in the paper for comparison.

Recently, some authors have proposed the application of GAs to solve some of the problems related to concrete production and delivery. For instance, [3] considers the problem of scheduling a single production plant in order to satisfy delivery time constraints. In particular, each load has to be delivered in a certain desired time, and a linear penalty proportional to the amount of earliness / tardiness with respect to this reference time is introduced. The authors propose two approaches, an exact method suitable only for very simple cases, and a GA for instances of more realistic size. The GA solves the problem by constructing an initial schedule that attempts to arrange the jobs so that they can be delivered as close as possible to the desired time. Since this initial schedule may encompass overlapping jobs that cannot be produced simultaneously, the algorithm searches for the subset of non-conflicting jobs that lead to the smallest value of the cost function. It should be noted that the paper considers only a single depot, does not consider limited resources for transportation, and considers instances that are significantly smaller than those used in our case studies.

Another recent research work on just in time cement production and delivery is reported in [1]. As in the previously mentioned research, the authors focus on scheduling a single depot, equipped with a fleet of vehicles with identical capacity and a fixed (customer and depot independent) loading/unloading times. Similarly to our study, they consider the case of orders exceeding the capacity of a single vehicle, which determine the need for a sequence of tightly coupled successive sub-deliveries operated by separate trucks. The authors propose the use of a GA for searching a production sequence that maximizes a predefined performance index (taking into account truck waiting times and a penalty for violating the unloading continuity of multi-truck orders) that is evaluated by discrete-event simulation of the operations of the fleet of vehicles. It is worth emphasizing that the assumption of identical production and loading/unloading times for each order is not acceptable in our real-world case studies. It also transforms the scheduling of production into a simpler job permutation / sequencing problem. Moreover, it should be noted that the largest instance considered in the paper (composed of a single depot processing 9 demands divided into 22 distinct sub-deliveries performed by a fleet of 20 trucks) is significantly smaller than the cases considered here (five depots receiving 40 to 70 demands, determining 200 to 450 sub-deliveries assigned to a fleet of about 50 trucks).

3 Modelling The Ready-Made Concrete Supply Chain

We now give a more detailed description of the concrete delivery problem examined in this paper. First of all, it should be mentioned that once the production center adds water to the mix of dry materials, the concrete has only about two hours (unless specific chemical retarders are employed) before the hydration process forms a gel that, if disrupted, would compromise the ultimate strength of the concrete. Thus, large orders require a strictly uninterrupted supply of concrete in order to avoid dangerous construction joints [15]. A delivery sequence for uninterrupted supply at a site requires a loaded truck to be available at the site when the preceding truck has ended the unloading. In some circumstances, there can be extremely large orders that involve a considerable number of trucks and thus tie up most of production plant's and vehicle fleets capacity. Since concrete should be placed no later than two hours after the addition of water, travel from the batch plant to a site should not take much more than an hour or so. Therefore, a plants operating radius tends to be limited based on the nature and condition of haul roads. The time a ready-mix truck may sit in traffic during rush hours is a significant consideration when scheduling site deliveries. On-time delivery of concrete is essential to a customer. If a truck arrives early, the concrete placement crew may not yet be ready. If a truck arrives late the continuity of the unloading is violated, and if the delay exceeds the concrete setting time the entire load has to be disposed.

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 the produced cement is loaded on the trucks for its delivery. Moreover, each loading dock can service only one truck at a time. Each PC can supply a variety of concrete types, that are obtained by mixing water and predefined amounts of dry material while they are being loaded on the truck. Thus, each order or fraction of an order is actually produced at the time the truck assigned to it is available at the loading dock for loading the material. As also remarked in [15], since cement mixing operation is highly automated, and dry components can be supplied at the loading dock in very short times, the production time is entirely determined by the time needed to place the load on the truck. 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. Moreover, there is no significant setup time depending on the type of concrete mixed in two successive load operations.

An order consists minimally of a delivery moment (date and time), type of concrete, required amount and delivery location. Additionally, a customer may require explicitly that an order be produced by a specific PC or that it must not be supplied by some of the PCs of the chain. Several other attributes of customers' resources at the unloading site must also be taken into account. Namely, each customer can unload at a maximum rate, and may specify a maximum amount of product per single delivery. Moreover, the customer may require a truck to arrive in advance with respect to the requested delivery time for some logistic reasons at the construction site. In other words, the customer may require that the truck waits a predefined time interval at the unloading site prior to begin unloading. Some orders do not need a delivery, as the customer picks up the concrete himself. Clearly, these orders must be taken into account only in the PCs scheduling.

Some characteristics of the trucks are also fundamental parameters of the whole scheduling problem. In fact, the delivering trucks have a limited capacity, and a maximum unloading rate. Some specific types of concrete require that a certain fraction of the trucks capacity is left empty, thus reducing the actual capacity of the truck for that type of cement. Trucks are parked in specific base locations at the PCs from which they start every morning, and to which they must return every evening. A vehicle can be used for a predefined day shift, so any delay or additional job must be paid additionally. Differently from many other routing problems, here each 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 only partially loaded during the delivery. As for the loading process, only one truck at a time can be unloaded at the delivery location (if two trucks have arrived at the location, one must wait until the other one is unloaded). Finally, whenever needed, it is possible to hire a number of additional vehicles from external companies.

We model the integrated supply chain scheduling as a cost minimization problem. For convenience, the list of all the symbols and acronyms used in our model is reported in Appendix A. The symbols are grouped according to their indices defined as follows. We suppose that at a given decision time, R (request) demands from different customers have been received, and have to be assigned to D (depot) different PCs. If a demand exceeds the capacity of a single truck, it is divided in a number of sub-demands (jobs), which will be delivered to customers. Thus, we introduce the following indices:

$d \in \{1, \ldots, D\}$**,** *depot-related index.*

$r \in \{1, \ldots, R\}$**,** *customer or demand-related index.*

$i \in \{1, \ldots, N\}$**,** *job-related index relative to the job.* N is the total number of jobs to perform. The sub-demands are arranged in sequential orders, so that the index i can be interpreted as follows:

$$i \in \left\{ \underbrace{1, \ldots, Z_1}_{r=1} \;\; \underbrace{Z_1 + 1, \ldots, Z_1 + Z_2}_{r=2} \;\; \underbrace{Z_1 + Z_2 + 1, \ldots, Z_3}_{r=3} \;\; \cdots \;\; \underbrace{\sum_{r=1}^{R-1} Z_r + 1, \ldots, N}_{r=R} \right\},$$

where Z_r is the number of sub-deliveries in which a request exceeding a truck's capacity is divided. Furthermore, we indicate with f_r and l_r the first and last job in the demand r, respectively.

$k \in \{1, \ldots, K\}$**,** *truck-related index.* K is the total number of trucks ($K = K_c + K_o$, where K_c is the number of trucks of the company, and K_o is the number of additionally hired trucks).

$m \in \{1, \ldots, M_k\}$**,** *task-related index.* A task of a truck is the delivery of a job to its destination. M_k is the maximum number of tasks allowed to a single truck k.

Preferably, the trucks owned by the cement delivery company are used. However, additional capacity (trucks) may be hired if necessary. If the D PCs are not able to supply the entire amount of requested materials, a part of the requests will be outsourced to external companies at an additional cost. We should mention that we assume that if the production of a job is outsourced, the job will be directly delivered to the customer at the specified time, i.e. our model does not deal with the delivery of outsourced production. On the contrary, we do consider the scheduling of hired trucks for the delivery of jobs that cannot be handled by the internal fleet as a part of our problem. Thus, our model considers the following decision variables:

$X_{ikm} \in \{0,1\}$	If the job i is assigned to truck k as m-th task, $X_{ikm} = 1$, otherwise $X_{ikm} = 0$.
$Y_{id} \in \{0,1\}$	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 $Y_{oi} = 0$.

The cost function is composed of three terms:

$$C = C' + C'' + C'''. \tag{1}$$

The first one, C', takes into account the transportation costs, in terms of total distance travelled by the fleet of trucks to deliver all the produced jobs (see (2)). This first term for transportation costs takes into account the sum of all the distances from PCs to customer sites. The second term for transportation costs accounts for the return trip to the PC for the next job, while the third term accounts for the cost of reaching the PC supplying the first job of the truck, if it differs from the base location, and the cost of returning to the base-location at the end of the working day. From a global viewpoint, the production of the entire supply chain should be organized so as to minimize the delivery costs

$$C' = \frac{CP}{V}\left(A_1 + A_2 + A_3\right), \tag{2}$$

where

$$A_1 = \sum_{i=1}^{N}\sum_{d=1}^{D} Y_{id}\Lambda(r_i, d), \tag{3}$$

$$A_2 = \sum_{m=1}^{M-1}\sum_{i=1}^{N}\sum_{\substack{j=1 \\ j \neq i}}^{N} X_{ikm}X_{jk(m+1)}\left(\sum_{d=1}^{D} Y_{jd}\Lambda(r_i, d)\right), \tag{4}$$

$$A_3 = \sum_{k=1}^{K} \left(\Lambda(DP_k, DF_k) + \Lambda(DL_k, DP_k)\right). \tag{5}$$

The second term C'' takes into account the loading and unloading waiting times (see (6)). As mentioned before, this is a critical variable of the problem. The waiting times measure how much in advance the truck must be ready for the operations at a PC or at a delivery site with respect to the actual start of the loading or unloading operation. In principle, waiting times should be minimized because they typically represent a loss (the more the waiting times, the lower the resource utilization). However, waiting times are the only safety margins of a given solution, since a schedule with reasonable waiting times allows a truck to perform its operation also in presence of delays (e.g. transport delays due to traffic).

$$C'' = CA\left(\sum_{k=1}^{K}\sum_{m=1}^{M} LWT_{km} + \sum_{i=1}^{N} UWT_i\right). \tag{6}$$

The third term C''' accounts for the additional costs related to outsourced production, hired trucks and the overtime work for some truck drivers (see (7)). 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. Of course, this contrasts with the just mentioned safety margins, because in general optimized schedules have very tight safety windows.

$$\begin{aligned} C''' = {} & PT\left(\sum_{i=1}^{N} Y_{oi}\frac{Q_{r(i)}}{Z_{r(i)}}\right) + HC\ K_o + \\ & + XTR\left(\sum_{k=1}^{K} \max\left\{0, STWD_k - T_k^{start}\right\} + \max\left\{0, T_{kM_k}^{6} - ENDWD_k\right\}\right). \end{aligned} \tag{7}$$

The optimization model is subject to a considerable number of assignment and timing constraints that will be introduced and commented separately in the next subsections.

3.1 Assignment constraints

Every job can be assigned to a production depot or outsourced once, and so $\forall i \in \{1,\ldots,N\}$,

$$\sum_{d=1}^{D} Y_{id} + Y_{oi} = 1. \tag{8}$$

Similarly, each job can only be assigned once to either a truck of our fleet or to an hired one, i.e. $\forall i \in \{1, \ldots, N\}$

$$\begin{cases} \text{if } Y_{oi} = 0, & \sum_{k=1}^{K} \sum_{m=1}^{M} X_{ikm} = 1 \\ \text{if } Y_{oi} = 1, & \sum_{k=1}^{K} \sum_{m=1}^{M} X_{ikm} = 0 \end{cases} . \tag{9}$$

Finally, jobs must be assigned to trucks sequentially (in other words guaranteeing that for the m-th task of any truck k all the preceding tasks are assigned and all the succeeding task are not assigned). $\forall k \in \{1, ldots, K\}, \forall m \in \{1, \ldots, M_k - 1\}$,

$$\sum_{i=1}^{N} X_{ik(m+1)} \leq \sum_{i=1}^{N} X_{ikm} \leq 1. \tag{10}$$

3.2 Computation of operation and travel times

The considered supply problem has a number of constraints related to some start or end times of specific truck operations. In order to clearly introduce such constraints, Fig. 1 defines the typical sequence of operations for a truck, and specifies the associated time intervals. In particular, once a truck is available at the PC, it has to

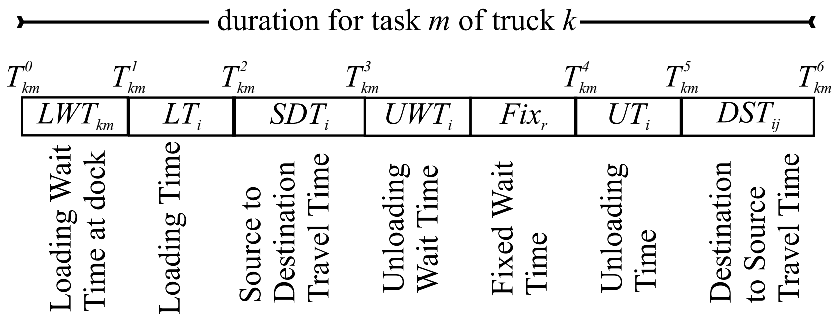
1. wait for the start of loading operations,
2. load the material,
3. travel to destination,
4. wait for unloading (including the customer-specified fixed time),
5. unload, and
6. return to a PC or base location.

The *job loading time* LT_i depends only on the depot loading rate LR_d and on job size Q_r/Z_r, i.e. $\forall i : f_r \leq i \leq l_r$

$$LT_i = \sum_{d=1}^{D} Y_{id} \left(FLT_d + \frac{Q_r}{Z_r} \frac{1}{LR_d} \right), \tag{11}$$

while the *source to destination travelling time* SDT_i for job i is computed as $\forall i : f_r \leq i \leq l_r$

$$STD_i = \frac{\sum_{d=1}^{D} Y_{id} \, \Lambda(r_i, d)}{V}. \tag{12}$$



T^0_{km} start of the task m of truck k

T^1_{km} start loading of task m of truck k

T^2_{km} end loading and start outward trip

T^3_{km} end outward trip and start of waiting for unloading

T^4_{km} end of waiting and start unloading

T^5_{km} end unloading and start trip toward the source of the next task

T^6_{km} end trip toward the source of the next task

Figure 1: Sequence of operations for a single truck.

Finally the travelling time between the destination of job i and the source of job j DST_{ij} is computed as follows: $\forall i : f_r \leq i \leq l_r$

$$DST_{ij} = \frac{\sum_{d=1}^{D} Y_{jd}\, \Lambda(r_i, d)}{V}. \tag{13}$$

3.3 Delivery time-window related constraints

Since each customer assigns the time window for delivery, a preliminary verification of data consistency should be carried on. In particular, it should be inspected if $\forall r\{1, \ldots, R\}$

$$LDT_r - EDT_r \geq Z_r\; UT_i, \tag{14}$$

i.e. that the customer assigned delivery time-window is sufficiently wide to allow the completion of unloading operations at the available unloading rate of the customer r. If this requirement is violated, a warning is issued to the customer. Moreover, accepted jobs must be scheduled meeting the following constraints.

$\forall i : f_r \leq i \leq l_r$

$$ELT_i - SLT_i \geq LT_i. \tag{15}$$

This condition guarantees that the time window allocated to each single job is large enough to allow the completion of the loading at the depot d.

$\forall i : f_r \leq i \leq l_r$

$$SLT_i \geq LDT_r - (l_r - i)\ UT_i - Tset_r. \tag{16}$$

This constraint guarantees that the start of loading time-window is not so early that the concrete may solidify before it is completely unloaded.

$\forall i : f_r \leq i \leq l_r$

$$ELT_i \leq LDT_r - SDT_i - Fix_r - (l_r - i + 1)\ UT_i. \tag{17}$$

This constraint affects the end of the loading time window, ensuring that the end of the loading cannot be so late that the sequence of remaining jobs will not be completed within the customer-specified time window.

$\forall d \in \{1, \ldots, D\}$, $\forall i_1, \forall i_2 \in \Gamma_d, i_1 \neq i_2$,

$$ELT_{i_1} \leq SLT_{i_2} \vee SLT_{i_1} \geq ELT_{i_2}. \tag{18}$$

This constraint forbids the overlap of loading time windows at the same depot.

3.4 Single truck related constraints

Truck operation times must be computed according to the following relations, by meeting the constraints introduced subsequently. $\forall k\{1, \ldots, K\}$

$$T^0_{k1} = \frac{\Lambda(DP_k, DF_k)}{V} + T^{start}_k. \tag{19}$$

The start of task m for truck k can be related to the time T^0_{k1} when it begins its first task as follows: $\forall k \in \{1, \ldots, K\}$, $\forall m \in \{2, \ldots, M_k\}$,

$$\begin{aligned} T^0_{km} &= T^0_{k1} + \sum_{l=1}^{m-1} \left(LWT_{kl} + \sum_{i=1}^{N} X_{ikl}\ (LT_i + SDT_i + UWT_i + FIX_i + UT_i) \right) + \\ &\quad + \sum_{l=1}^{m-1} \sum_{i=1}^{N} \sum_{\substack{j=1 \\ j \neq i}}^{N} X_{ikl}\ X_{jk(l+1)}\ DST_{ij}. \end{aligned} \tag{20}$$

$\forall k \in \{1, \ldots, K\}$,

$$LWT_{k1} = T^1_{k1} - T^1_{k0}. \tag{21}$$

$\forall k \in \{1, \ldots, K\}, \forall m \in \{2, \ldots, M_k\}$

$$LWT_{km} = T^1_{mk} - \sum_{i=1}^{N} X_{ik(m-1)}\, T^6_{k(m-1)}. \tag{22}$$

$\forall k \in \{1, \ldots, K\}, \forall m \in \{1, \ldots, M_k\}$,

$$\begin{aligned} LWT_{km} &\geq MWT, & (23)\\ T^1_{km} &= T^0_{km} + LWT_{km}, & (24)\\ T^2_{km} &= T^1_{km} + \sum_{i=1}^{N} X_{ikm}\, LT_i, & (25)\\ T^3_{km} &= T^2_{km} + \sum_{i=1}^{N} X_{ikm}\, SDT_i. & (26) \end{aligned}$$

$\forall i = f_r$

$$UWT_i = \max\left(EDT_r - Fix_r - \sum_{k=1}^{K}\sum_{m=1}^{M} X_{ikm}\, T^3_{km}, MWT\right). \tag{27}$$

This condition states that for the first job related to a demand r, the truck will have to wait longer than MWT only if it arrives earlier than the expected T^3_{km}.

$\forall i : f_r < i \leq l_r$,

$$UWT_i = \max\left(\sum_{m=1}^{M}\sum_{k=1}^{K} X_{(i-1)km}\, T^5_{km} - Fix_r - \sum_{k=1}^{K}\sum_{m=1}^{M} X_{ikm}\, T^3_{km}, MWT\right). \tag{28}$$

Analogously, this condition specifies that for the all the other jobs related to a demand r, the truck will have to wait longer than MWT only if it arrives earlier than the end of the unloading of the previous job.

$\forall k \in \{1, \ldots, K\}, \forall m \in \{1, \ldots, M_k\}$,

$$\begin{aligned} T^4_{km} &= T^3_{km} + \sum_{i=1}^{N} X_{ikm}\, \left(UWT_i + Fix_{r_i}\right), & (29)\\ T^5_{km} &= T^4_{km} + \sum_{i=1}^{N} X_{ikm}\, UT_i, & (30)\\ T^6_{km} &= T^5_{km} + \sum_{i=1}^{N} \sum_{\substack{j=1\\ j \neq i}}^{N} X_{ikm}\, X_{jk(m+1)}\, DST_{ij}. & (31) \end{aligned}$$

$\forall k \in \{1, \ldots, K\}, \forall m \in \{1, \ldots, M_k - 1\}$,

$$T^6_{km} = T^0_{k(m+1)}. \tag{32}$$

$\forall k \in \{1, \ldots K\}$,

$$T^6_{kM_k} = \frac{\Lambda(DL_k, DP_k)}{V} + T^5_{kM_k}. \tag{33}$$

The latter equation simply models the return of each truck to its base location.

Truck scheduling constraints are defined as follows. $\forall i : f_r \le i \le l_r$,

$$SLT_i \le \sum_{k=1}^{K} \sum_{m=1}^{M} X_{ikm} \, T^1_{km} \le ELT_i - LT_i = LLT_i. \tag{34}$$

This means that the loading of job i must start and end within its loading time window.

$\forall i : f_r \le i \le l_r$,

$$\sum_{k=1}^{K} \sum_{m=1}^{M} X_{(i+1)km} \, T^1_{km} \ge \sum_{k=1}^{K} \sum_{m=1}^{M} X_{ikm} \, T^6_{km} + MWT. \tag{35}$$

This constraint guarantees that T^1_{km} is chosen so as to ensure that the LWT_{km} is greater than the minimum allowed safety margin MWT.

$\forall i : f_r \le i \le l_r$,

$$\sum_{k=1}^{K} \sum_{m=1}^{M} X_{ikm} \, T^1_{km} \le \sum_{k=1}^{K} \sum_{m=1}^{M} X_{ikm} \, T^4_{km} - SDT_i - Fix_r - MWT. \tag{36}$$

Similarly, this constraint means that the choice of T^1_{km} must guarantee that UWT_i is not less than MWT.

$\forall i : f_r \le i \le l_r$,

$$\begin{array}{l} \sum_{d=1}^{D} \sum_{j=f_r}^{i-1} Y_{jd} = 0, \text{ and } \sum_{d=1}^{D} Y_{id} = 1 \\ \qquad EDT_r + (i - f_r)\; UT_i \le \sum_{k=1}^{K} \sum_{m=1}^{M} X_{ikm} \, T^4_{km} \le LDT_r - (l_r - i + 1)\; UT_i. \end{array} \tag{37}$$

This equation considers the case in which some of the jobs composing a demand are outsourced. In particular, the equation refers to the case in which the first jobs in the sequence (all those preceding job i) are outsourced. In such a case, the job i must be scheduled so that the preceding jobs are fully unloaded. All the other cases are considered in other equations (see (38)).

3.5 Constraints between different trucks

The unloading of all the jobs (but the first one) composing demand r must start exactly when the preceding one ends. This strict requirement guarantees the continuity of the unloading process that is fundamental in many construction sites. The model takes into account this requirement with the following constraint: $\forall i : f_r \leq i < l_r$,

$$\sum_{k=1}^{K}\sum_{m=1}^{M} X_{ikm}\, T^5_{km} = \sum_{k=1}^{K}\sum_{m=1}^{M} X_{(i+1)km}\, T^4_{km}. \tag{38}$$

4 Hybrid Metaheuristics For Concrete Production And Delivery

Given a certain set of demands, finding a feasible schedule for the overall supply chain may be a task of variable but generally prohibitive complexity, which depends not only on the number but also on the specific characteristics of the demands. For instance, it can happen that the requests for a given day are well within the overall capacity of the network, but concentrated in areas that are out of the operating radius of most PCs, which consequently remain inactive. Similarly, requests may be conflicting in terms of closeness of their delivery times, so that it is not possible to schedule the loading operation on the various PCs guaranteeing the delivery of all the jobs in the requested time-windows. Many other problems may arise due to the limited fleet of vehicles for delivery. Moreover, there are a number of practical requirements that have not been explicitly taken into account in the proposed model for sake of simplicity, but may have a significant influence on the quality of the final solution. For instance, it is preferable to make all the efforts to keep the sub-demands composing a single request assigned to the same PC, in order to avoid schedules with an excessive coupling between the PCs. Moreover, some customer may explicitly require the production at a specific PC, thus constraining some of the decision variables Y_{id} to predefined values. In order to take into account all the requirements of the problem, we have devised a heuristic algorithm that decomposes the supply-chain problem in two separated sub-problems, which are tackled one after the other. In particular, in a preliminary stage, the demands are subdivided into separated jobs according to trucks capacity and customers requirements. Then, the first problem regards the assignment of jobs to PCs, and the associated scheduling of the simultaneous mixing and loading operations at each PC. When a loading operation is scheduled at a PC, it is assumed that a truck will be made available at the loading dock at the assigned loading start time. The second problem regards the routing of the fleet of carriers, guaranteeing that a truck is located at a loading dock at a time a loading operation is scheduled to start. If the routing algorithm is not able to make a truck be available in time for the loading of the assigned job, an external truck is hired for this purpose at an additional cost. Clearly, this decomposition of the problem in two consecutive problems may lead to sub-optimal solutions, while ideally the problems of PC scheduling and truck routing should be jointly considered in a single global formulation. However, the decomposition allows us to achieve two subproblems of reasonable complexity, and solve them effectively with relatively simpler algorithms. Moreover, it should also be remarked that the experimental investigation in our case study shows that the proposed search strategy is in general able to find solutions with generally short and evenly distributed waiting times, confirming that the internal fleet of trucks is in general appropriately exploited.

The two separated sub-problems are solved with different heuristic algorithms. Namely, a GA is used to optimize the assignment of demands to depots (the decision variables Y_{id}) and the order of priority of loading of the demands to the assigned depots, while efficient constructive heuristics are used to deal with timing constraint satisfaction (i.e. determining a feasible overall schedule according to the assignment performed by the GA) and truck dispatching. For the sake of clarity, we describe the GA and the constructive heuristics in separate subsections.

4.1 The Genetic Algorithm

Genetic algorithms belong to a class of stochastic search methods that work iteratively on a population of candidate solutions of the problem (individuals), performing a search guided by the fitness (i.e. the value of the objective function) of each solution. In particular, the higher the fitness, the more the genes of a solution are likely to be propagated to the solutions explored in the next iterations. This Darwinian principle is emulated with specific crossover, mutation and selection operators, which are applied with stochastic mechanisms that make the GA explore solutions with increasing fitness. One of the frequently acknowledged merit of these optimization algorithms is their flexibility with regards to the characteristics of the objective function, as they do not rely on specific a priori hypotheses (e.g. continuity and convexity).

Every GA requires a preliminary definition of an encoding strategy to transform a generic solution of the problem into a string of symbols, chosen from a pre-specified alphabet and suitable to the application of recombination operators for generation of new solutions (i.e. crossover and mutation operators). In GAs literature, an encoded solution is generally referred to as chromosome, and a single parameter of the solution vector is called a gene. As mentioned, an ex-

Customer's Request-to-Depot Assignment						Priority of request in schedule construction					
r_1	r_2	r_3	r_4	r_5	r_6	p_1	p_2	p_3	p_4	p_5	p_6
1	3	2	1	2	2	4	5	6	1	3	2

Figure 2: A generic chromosome.

tremely rich literature on GA-based sequencing and scheduling methods is available to date, and a considerable variety of different encoding strategies have been devised to address specific requirements. When dealing with scheduling problems, there are several conflicting requirements in the definition of the coding strategy. In particular, the coding strategy should be devised so as to have compact chromosomes that completely characterize the associated solution. Ideally, the coding should be defined so as to ensure that the crossover (mutation) of two (one) chromosomes describing legal (i.e. satisfying all the constraints) solutions always lead to legal new solutions. For complex scheduling problems, such as the one considered here, guaranteeing legal offspring involves either extremely long chromosomes (which may compromise the search ability of the GA) or very complex and computationally demanding crossover and mutation operators. Many different solutions to overcome these problems in related contexts have been proposed in the literature (e.g. [6], [8], [11], [14], [18] to mention only some). Our research tackles the solution representation problem by adopting a coding strategy that describes only a part of the whole scheduling problem, and then using constructive heuristics to determine the complete schedule associated with the chromosome each time the fitness of a chromosome is computed.

To illustrate the basic mechanisms of our strategy, let us first focus on the chromosome encoding. The chromosome is made up by two separate parts, both containing R (the number of demands) elements, as described in Fig. 2 for $R = 6$. The first part defines the assignment of demands (requests) to the depots. Each gene is an integer number between 1 and D (the number of depots): the lth gene of this first part of the chromosome indicates the depot to which request r_l is assigned. For instance, in the chromosome represented in Fig. 2, requests r_1 and r_4 are assigned to depot 1, r_2 to depot 3 and all the remaining ones to depot 2. The second part of the chromosome establishes the order in which the R requests will be considered in the construction of the complete schedule of the production chain. The l-th gene in this second part indicates the demand that will be considered at the l-th step of the scheduling construction

```
/* Single Criterion GA */
/* Algorithm Startup */
i = 1;
Pop(1) = random_pop
fitness_eval(Pop(1))
i = 2;
/* main loop of the GA */
WHILE terminating_condition == false
    p_best = findbest(Pop(i-1)) /* elitist preservation of
                                         the best-known individual*/
    Pop(i) = select(Pop(i-1),sel_ops);
    Pop(i) = crossover(Pop(i));
    Pop(i) = mutation(Pop(i));
    Fitness_eval(Pop(i))
    Pop(i) = Pop(i) ∪ p_best
    i = i + 1;
END WHILE
```

Figure 3: The basic structure of the GA.

procedure. For instance, in the chromosome in Fig. 2, request r_4 appears in the first location of the second part of the chromosome (p_1), and so it is assigned the highest priority. Thus, r_4 will be the first one to be allocated in the scheduling plan. Following the same sequence of appearance in the chromosome, the next demand in order of priority is r_5 (assigned with priority p_2), followed by request r_6 (priority p_3), and then, r_1, r_3, and finally r_2. Clearly, the second part of the chromosome can be any permutation of the sequence of integers $1, 2, \ldots, R$. In other words, the first part of the chromosome defines the values of the decision variable Y_{id}, while assignment variables Y_{oi} and X_{ikm} are not specified, but computed by the constructive heuristic procedure (described in Section 4.2), which is called every time the fitness of a new chromosome must be computed. The main schema of the GA is summarized in Fig. 3 using a descriptive meta-language. The set `Pop(`i`)` represents the set of solutions (including the associated fitness) composing the population at i-th iteration (generation). The role of each meta-function used in the description can be summarized as follows.

`random_pop` This function creates a random initial population of 100 individuals in our experiments.

`fitness_eval(Pop)` The fitness of each *new* individual (i.e. resulting from a random initialization, or from a crossover or mutation operation that produced a solution differing from its parent(s)) in `Pop` is computed. As mentioned, the chromosome of an individual only specifies a part of the decision variables of the scheduling process (Y_{id}), while the remaining variables (Y_{oi}, X_{ikm}) and the scheduled times for each operation are determined by a sequence of constructive heuristic algorithms that will be described in the next subsection. Once the overall schedule of the whole supply chain has been defined by the constructive algorithms, the value of the cost function associated to the individual is computed and assigned as fitness of the chromosome.

`select(Pop,sel_ops)` This function returns a new population of solutions selected from those in `Pop` with a strategy that assigns higher probability of selection to individuals with higher fitness. We use *tournament selection* [11] with two individuals for each tournament.

`crossover(Pop)` This function randomly selects couples of solutions in `Pop` to perform a crossover that returns two new individuals, which partially inherit some characteristics of both parents. After the crossover, the resulting offspring replaces the two parents. Given the particular structure of our the chromosome structure we devised a new operator that combines two effective operators derived from recent literature (e.g. [13], [4]). Our crossover randomly selects a chromosome cut point. If this point falls on the first half of the chromosome, it performs a standard single-cut crossover to the first part of the chromosome (request to depot assignment), otherwise it performs an order-based crossover on the remaining part. The effects of this operator are illustrated in Fig. 4.

`mutation(Pop)` This function randomly alters a solution to obtain a new one. 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

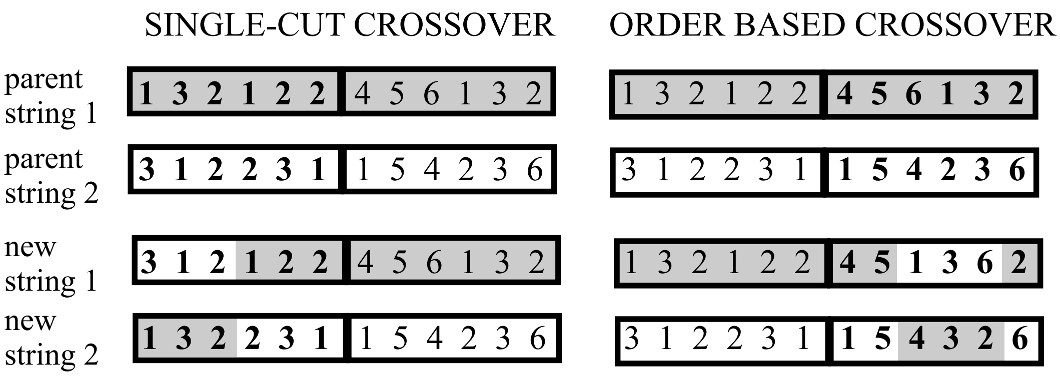


Figure 4: Examples of the crossover operator.

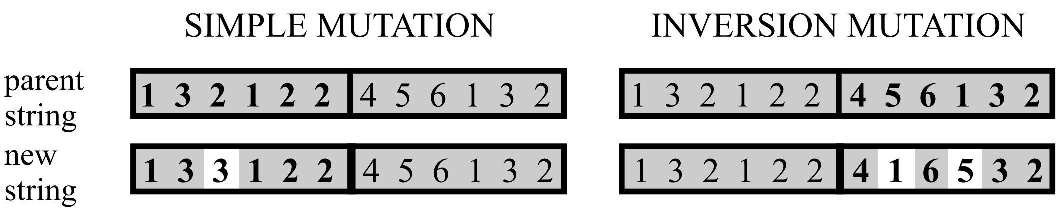


Figure 5: Examples of the mutation operator.

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. The effects of this operator are illustrated in Fig. 5.

`terminating_condition` For all our experiments, we stopped the algorithm after 200 generations.

Once a part of the solution is specified in the chromosome, a constructive heuristic procedure (CHP) is used to determine a legal schedule for the entire supply chain. The CHP is composed of two separate parts, respectively dealing with the scheduling of loading operations at the PCs, and the scheduling of job deliveries by trucks. For the sake of brevity, it is not possible to provide an exhaustive description of all the steps performed by each part of the CHP, so in the following we focus on the illustration of the main mechanisms of each part.

4.2 Constructive Heuristic Procedure: scheduling loading operations

Basically, the procedure starts to process the demands following the order of priority specified in the second part of the chromosome. The operations of the CHP are relatively simple when there is only one job for each request ($R = N$). If this is the case, the first processed demand is scheduled on the PC assigned in the chromosome, unless the PC-to-customer distance, PC loading rate, and customer unloading rate make this assignment unfeasible. In the latter case, the demand is redirected to the nearest depot to the delivery location of the demand. When no conflict with previously scheduled demands is detected, a demand r is scheduled so that the unloading (of its first job) starts exactly at the EDT_r. The second and successive demands are scheduled with similar criteria. Firstly, it is inspected if they can be scheduled at the assigned PC so that they are delivered and unloaded within the specified time window. When a demand r is scheduled at a PC that has already been assigned other demands, some different situations can occur:

1. the customer-specified time-windows for the requests are such that the production times of r is not in conflict with the previously assigned ones, or

2. r is in conflict with some other job loading at the PC, and the assignment cannot be accepted as is.

The partial overlap of loading times is one of the simplest conflicts that may arise in the complex scheduling problem addressed here. The CHP firstly tries to shift forward the start of the loading time-window of r until it is no longer overlapping with the others assigned to the same depot. Of course, this operation may have negative effects. In particular, the start time of unloading of r will be shifted forward as well and consequently it may happen that the delivery cannot be completed in the time requested by the customer. In this case the CHP makes a second adjustment trial, this time shifting backward the loading window of r until there is no overlapping. This operation is in general less favorable than the previous one because it implies that the truck will arrive at the delivery location before the EDT, and thus it will have to wait. The waiting time cannot be so long that the concrete starts to set before its complete unloading. If this constraint is violated, there is no other solution except for either reassigning the demand to another PC, or making other adjustments that involve also the already assigned demands, i.e. those having a higher priority than the one of r. We must note that whenever a request is reassigned to a depot differing from the one specified in the chromosome, the CHP will change the chromosome accordingly. If a request cannot be successfully reassigned to one of the available depots of the company, then it is outsourced. Thus, the priority of demands in schedule construction strongly influences the way the demand are scheduled over time. The

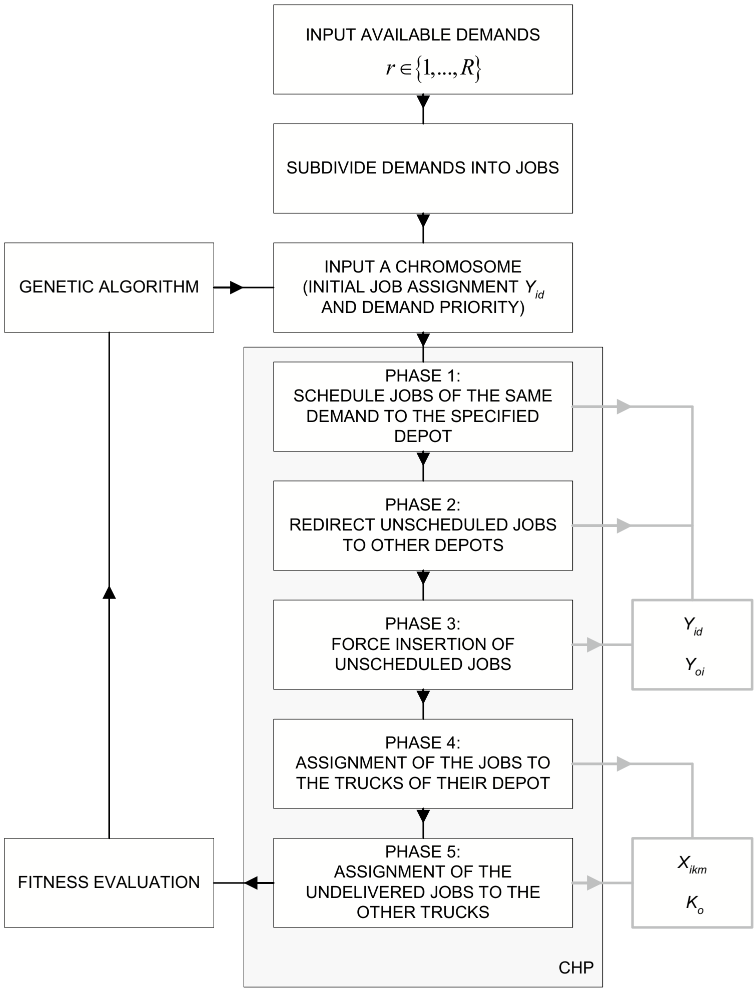


Figure 6: The main steps of the CHP and its integration in the GA.

way schedules are (re)constructed to meet all the timing constraints also makes possible that two or more different chromosomes lead to the same final schedule, and thus to the same fitness value.

In the most general case, each demand is composed of several jobs. In this case, the CHP has to perform a considerably larger number of operations. The general sequence of operations performed by the CHP is summarized in Fig. 6. Also in this case the CHP examines each demand in the chromosome in order of the assigned priority. Let us focus on a given demand r: after checking for possible infeasibility of the assignment due to excessive distance between delivery location and assigned PC (if it occurs, it is handled as described before), the algorithm starts to examine and schedule each job composing the demand. The algorithm computes the SLT of the first job of the demand based on the customer-specified EDT and of the company-specified UWT, avoiding the overlapping of the loading windows as explained in the previous section. If one of the adjustments leads to a successful schedule, then the gene specifying the assignment of the demand to the PC is confirmed and marked as unchangeable. Otherwise, the algorithm proceeds by attempting to assign the second job of the demand to the PC, using almost the same procedure described above. It is worth noticing that when the CHP operates on the second or subsequent jobs of a demand, it has to verify that both the preceding but not yet assigned jobs, and the following ones can be scheduled satisfying the delivery time 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 the demand can be scheduled on the PC. In the latter case (100% jobs must be reassigned elsewhere), the gene of the chromosome is actually changed and the procedure will start investigating the other PCs in order of shortest distance from depot to customers site.

After successfully assigning one of the jobs of the demand to a given PC, say d, the CHP proceeds to assign all the remaining jobs to the same PC d, guaranteeing that each job will be delivered so that the end of its unloading coincides with the start of unloading of the following job (see constraint (38)). In a second step, the CHP reconsiders the unscheduled jobs and tries to place them on other PCs, considered in order of increasing distance from the customers site. Note that it can occur that a job of a demand can be allocated in time earlier than other jobs of the same demand having a smaller index. The CHP will then sort the jobs of the same demand so that they have an increasing index. Moreover, this reorganization of the sequence of jobs evenly redistributes the unloading waiting times of all the involved jobs, thus leading to an improved schedule.

At this point, the main task of the CHP is to handle the jobs still not assigned to any PC. The final attempt to assign a job to a PC is performed as follows. Starting from the first PC in order of increasing distance to customers site, the CHP tries to "force" the insertion of the job at the exact time that guarantees the ideal unloading time. The main property of this last attempt is that the CHP now is allowed to shift-backward also the jobs already scheduled in previous steps (including those having an higher construction priority). This operation is likely to have a significant impact on the overall schedule at the PC, and also to increase the value of the cost function associated to the solution. To evaluate the actual advantages of this insertion, the CHP computes the increment of cost caused by the insertion of the job i in the schedule. If either the increment exceeds the cost of outsourcing of the job i, or the insertion causes a violation of some feasibility constraint, the insertion is rejected and the CHP tries with the next PC. On the contrary, if the insertion determines a cost increment that is lower than the cost of outsourcing, then the insertion is accepted, and the schedule is modified accordingly.

4.3 Constructive Heuristic Procedure: scheduling trucks

Once the assignment solution encoded in the chromosome is converted in a feasible loading sequence for each PC, the fleet of trucks must be assigned to jobs (setting the values of decision variables X_{ikm}) and routed from PCs to customer sites and vice-versa to pickup and deliver loads. Basically, the truck scheduling must guarantee that a truck assigned to a job is available at the loading dock of the supplying PC at the scheduled load start time. A heuristic procedure, referred to as *Truck Schedule Construction Algorithm* (TSCA) is in charge of performing this task. Initially, all the jobs that are marked as directly picked-up by customer-owned trucks are removed from the assignment list. Also the TSCA works in two consecutive and separated phases: first, it assigns the jobs produced at a given PC to the fleet of vehicles already owned by and located at the same PC, and second it searches for vehicles for delivering the remaining unassigned jobs. The main operations of the two phases are summarized in the flow charts shown in Fig. 7.

To illustrate the allocation procedure, let us firstly define the set of 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 (hereinafter defined as type 1) or have already completed some transport operations and can return to the PC before time t (type 2). This distinction is particularly relevant, since when both types of trucks are available, the TSCA always tries to assign those of type 2 first, in order to actually use the minimal amount of trucks for servicing all the requests. If the set of available trucks contains some trucks of type 1 over the whole working day, this clearly indicates that the size of the fleet exceeds the actual requirement. Now let us focus on the assignment mechanisms.

At the beginning of the working day $t = t0$, all the trucks of the PC are idle and ready for operation (all the trucks are of type 1). The TSCA allocates the first jobs produced in the working day to trucks of type 1 until the first truck of type 2 becomes available. From that time on, the TSCA always gives higher priority to trucks of type 2. When multiple trucks of type 2 are available, the TSCA ranks them in order of increasing return time (the time at which they are

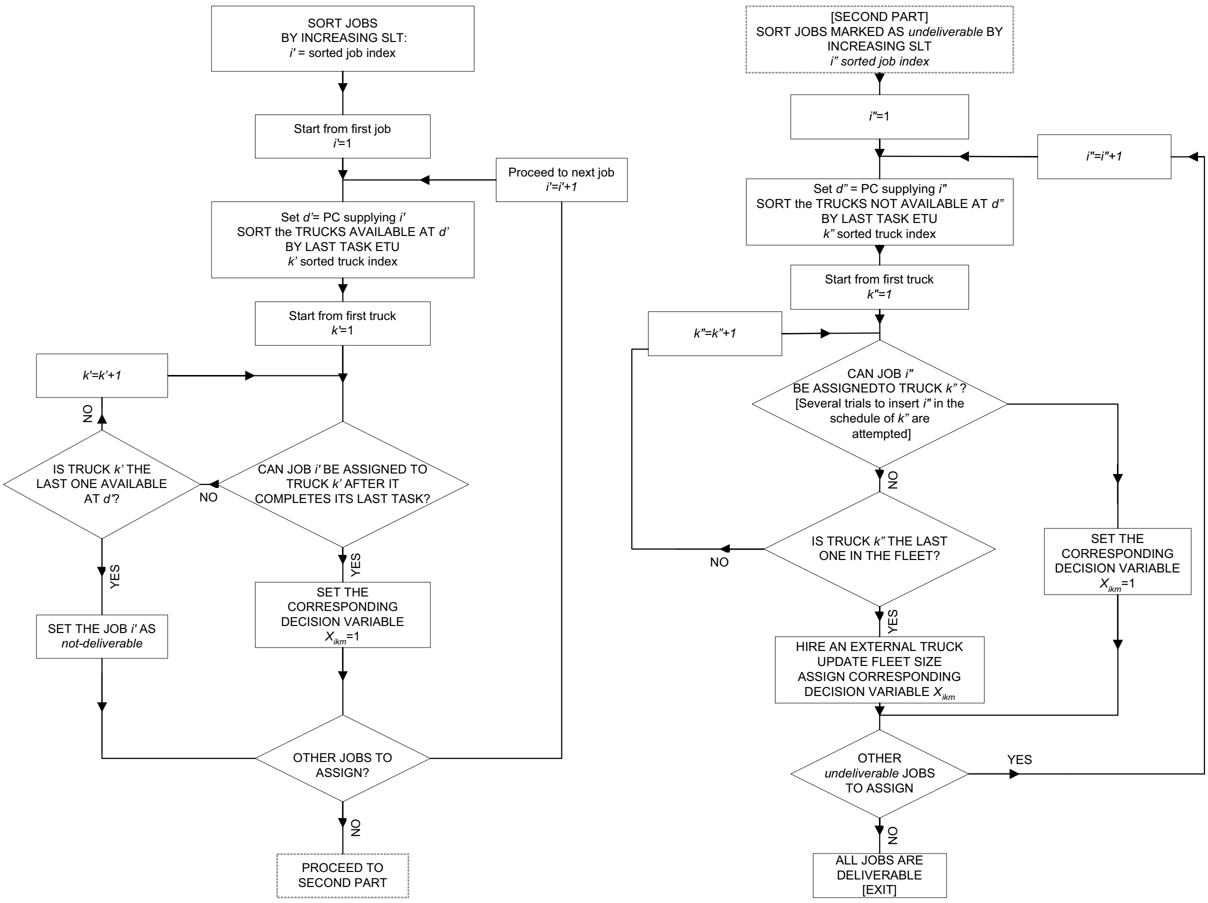


Figure 7: The truck schedule construction algorithm.

expected to be back at the base PC) and assigns the last one in the rank (the one with the latest return time) first. Therefore, the truck assignment strategy is also referred in this manuscript 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 such a priority strategy for the trucks is twofold. Firstly, as mentioned earlier, this strategy tends to use a minimal number of trucks in the assignment. Secondly, instead of evenly distributing the idle times among trucks, the SIT strategy causes some trucks to have longer idle times between assigned services. In this way, these trucks can be profitably assigned to jobs of other PCs, as done in the second part of the TSCA. When a job is assigned to a truck, the variable X_{ikm} is updated accordingly. When the first assignment strategy is unable to find a truck for a given job, the job is temporarily marked as *undeliverable*, and its assignment is postponed to the second part of the procedure. The first part of the TSCA proceeds with the job-to-truck assignment until it has inspected all the jobs. At this point, either all the jobs have been successfully assigned to the trucks of their respective supplying PCs, or there is a set of unassigned jobs marked as undeliverable that still needs to be handled. To sum up, the first phase of the TSCA attempts to assign the jobs of a PC to the smallest number of trucks already located at the PC.

The second phase of the TSCA involves the jobs marked as undeliverable. The main steps of this second part are summarized in the right-hand part of Fig. 7. As mentioned, jobs may be undeliverable because either they are scheduled on a PC that is not equipped with delivery vehicles (and explicitly relies on the support of trucks from other PCs), or the trucks of the PC have already been assigned to other delivery operations. In the first part of this second phase, the sets of unassigned jobs at each PC are merged and sorted in the order of increasing SLT. Let us focus on the first one of the resulting list, say job i'', and let us call d'' the PC supplying the job. The TSCA considers the set of remaining trucks sorted by completion time of last operation, and tries to assign i'' to the first truck in this list, say k''. To add i'' to the schedule of k'', the TSCA inspects 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 i'' cannot be assigned to any truck in the list of remaining ones, then a request for hiring an external truck is issued. In conclusion, at the end of TSCA, all the decision variables (Y_{id} , Y_{oi} and X_{ikm}) are assigned and the resources are scheduled so as to meet all the problem constraints.

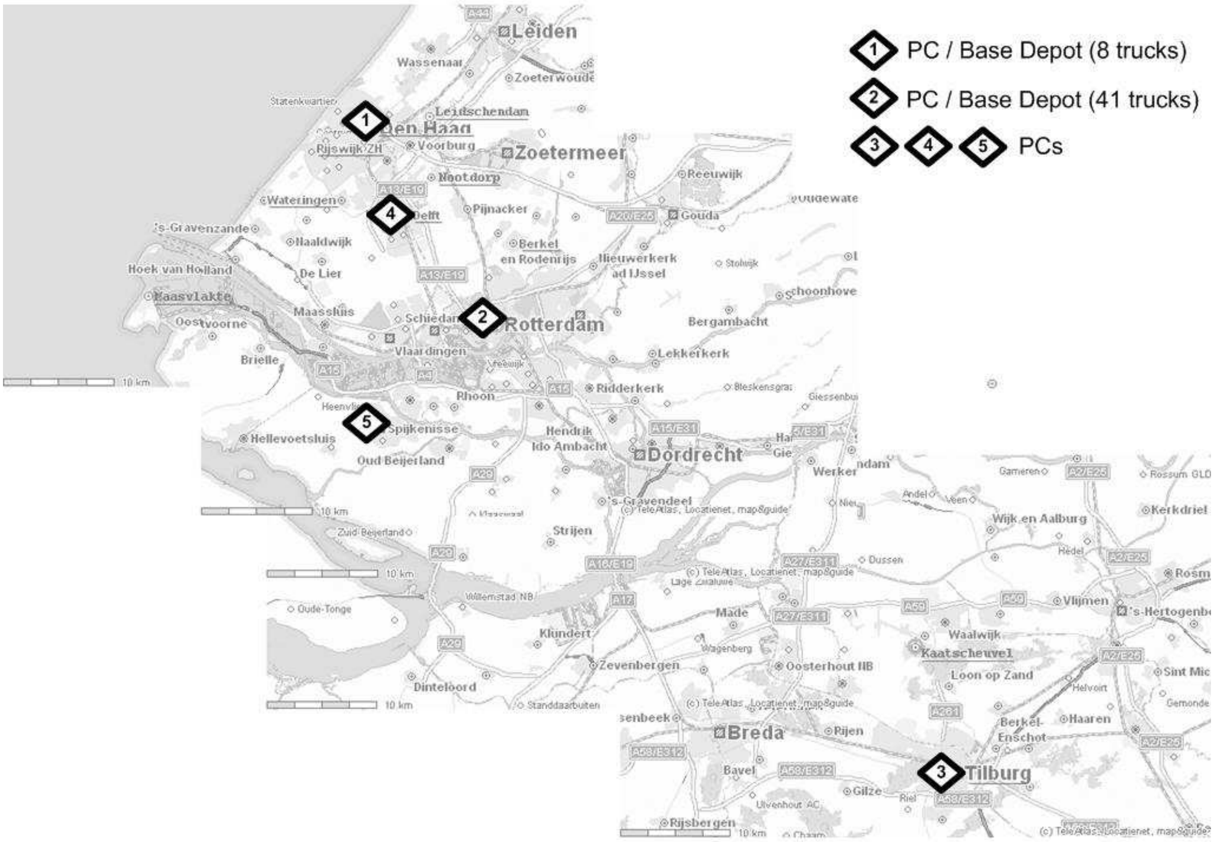


Figure 8: PCs locations.

5 Case Study

Our research work is based on a supply chain composed of five PCs located in the Netherlands. The fleet of trucks consists of 49 vehicles housed in two PCs. As we can see from Fig. 8, most of the PCs of this company are located around the Rotterdam port area. Though the location of the PCs is strategically planned, note that there are only two base depots for the trucks. This means that three PCs will have to rely on the other two for delivering their produced concrete. Our investigation focuses on information regarding a typical working day, with several requests from various customers spread over a large area surrounding the supply chain. Based on the available data, we generated a further set of 250 hypothetical instances having extremely variable characteristics (number and size of requests, concentration of requests in specific areas, conflicting time constraints, etc.), in order to evaluate the effectiveness of the proposed approach in a wide range of differing operating scenarios. It is worth mentioning that some characteristics derived from the analysis of the available demand patterns are noticeably close to those discussed in

Table 1: Weights in CU (Cost Units) for the components of the cost function.

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

related literature (high density of demand between 7:00-9:00 and 13:00-15:00, normal distribution for morning orders, exponential distribution for afternoon orders [10]). Table 1 summarizes the values of the main cost parameters in normalized cost units (CU).

Each considered instance differs in the number and characteristics of the requests. In particular, each demand specifies a delivery time window (EDT, LDT), a quantity Q required (in m^3), a maximal delivery size Mds, a fixed waiting time (Fix), an unloading rate (UR) and a percent of the truck that must be left empty (Per). The trucks have a maximal capacity C_{max} of 10 m^3. In general, customer requests have very narrow time windows, which imposes to schedule the delivery of the first job very close to the EDT. The average truck speed used in our model is $60\ \mathrm{Km/h}$ while the concrete setting time T_{set} 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 return to base location later than the end of the working day, an additional cost is incurred.

The prototype of our GA-based hybrid scheduling strategy was developed in Matlab mathematical programming environment. An execution of a single run of the GA configured as described in the previous section takes approximately 6 minutes on a Pentium 4 CPU 2.6 GHz. Although execution times could be dramatically reduced by translating the prototype algorithm into more efficient programming environments (e.g. C code), it is worth noticing that even the average execution time of the Matlab code is short enough to allow a quasi-realtime rescheduling in case a new urgent request is received while the current workplan is already started.

To test the effectiveness of the proposed hybrid GA approach, we compare it with four different scheduling policies obtained by applying assignment criteria that are suggested by experts as main criteria to build their schedules. Namely, the typical decision criteria in this context assign service priorities based on either the distance of the customers site from the nearest PC in the chain, or the size of the requests (higher priority to larger orders). Analogously, also trucks are generally assigned using a constructive procedure that takes into account distances and truck idle times. The four policies used here for comparison are obtained by the following different combinations of these heuristic rules.

a)*SD/SIT (Shortest Distance/Shortest Idle Time):* This heuristic criterion firstly sorts requests by decreasing size. Then, it starts allocating each job to the PC that is closest to job delivery location. Conflict and timing constraints are handled with adjusting procedures that are analogous to those illustrated in the previous section. When a job cannot be assigned to the nearest PC, this heuristic algorithm retries with the next PC in the order of increasing distance from delivery location (shortest distance). If no PC can supply the job, it is outsourced. Once all the jobs are scheduled, they are assigned to trucks. The strategy for truck assignment attempts to load each job on the truck that has been idle for the shortest time (shortest idle time).

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

c) *SW/SIT (Smallest Workload/ Shortest Idle Time):* After sorting requests by decreasing size, this heuristic algorithm tries to assign each job to PCs considering them in order of increasing (already assigned) workload (smallest workload). The truck assignment strategy is the same of the case a).

d) *SW/LIT (Smallest Workload/ Longest Idle Time):* This is a variant of policy c) 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 considered policies. This case considers 71 demands for a total amount of $2116.3\ \mathrm{m}^3$ of ready-made concrete, divided in 258 jobs. Timing details of each demand are summarized in Fig. 9. 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 customers. The total cost of the solution obtained by the GA is about 20% lower than the one provided by

Table 2: Summary of the results on a reference instance.

	Variant									
	SD/SIT		SD/LIT		SW/SIT		SW/LIT		GA	
components	value	cost	value	cost	value	cost	value	cost	value	cost
Jobs outsourced	3 ($28\ \mathrm{m}^3$)	56000	3 ($28\ \mathrm{m}^3$)	56000	6 ($48\ \mathrm{m}^3$)	96000	6 ($48\ \mathrm{m}^3$)	96000	0 ($0\ \mathrm{m}^3$)	0
Hired trucks	17	170000	22	220000	21	220000	23	230000	15	150000
Extra pay (min)	0	0	0	0	23	115	23	115	55	275
Empty trips (Km)	5290	52900	5119	51190	7123	71230	7242	72420	5479	54790
Loaded trips (Km)	3129	31290	3129	31290	6716	67160	6719	67190	5068	50680
Waiting time (min)	7214	108210	15315	229725	4803	72045	10657	159855	4665	69975
Total cost	417952 CU		587757 CU		515856 CU		624856 CU		325720 CU	

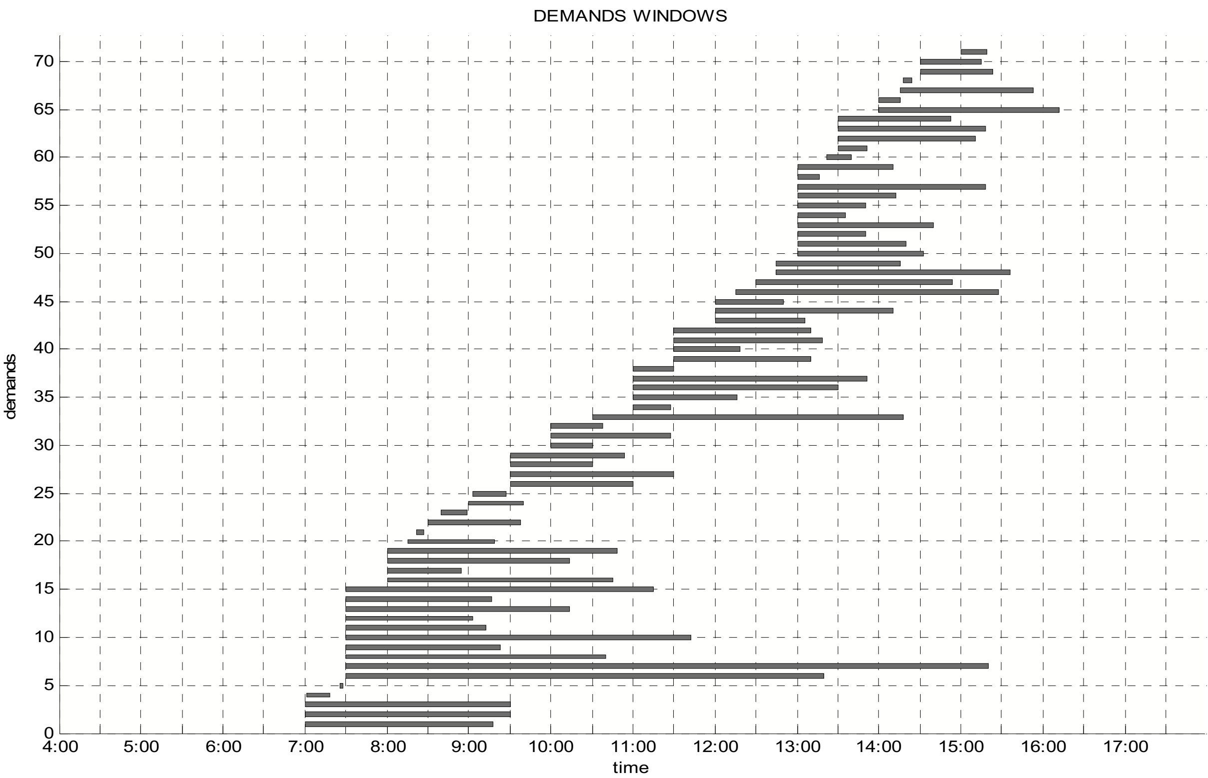


Figure 9: Customers' specified delivery windows.

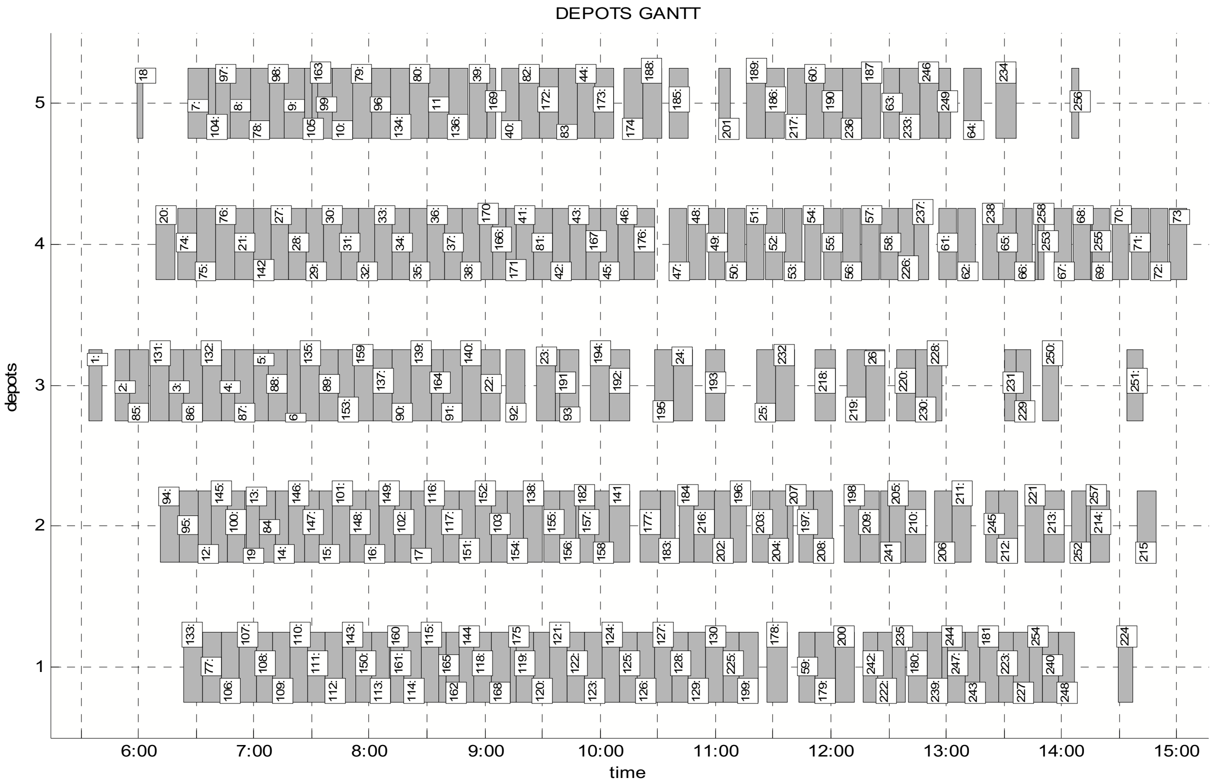


Figure 10: Chart of loading operations at the PCs.

SD/SIT, which is the most effective one amongst the heuristics compared with the GA. In particular, being 25 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 requests to the 5 PCs of the supply chain. The Gantt charts of loading and transportation operations corresponding to the solution obtained with the GA are shown in Fig. 10, Fig. 11 and Fig. 12.

Almost one third of the requests (including most of the larger ones) have their EDT between the 7:00 and the 9:00 AM (Fig. 9). This high concentration of demands requires that all the PCs contribute to the production. In this time interval, the effort of all the PCs is clearly visible in Fig. 10, which reports the Gantt chart of the loading operations at the five PCs. It can be noted that between 6:30 and 9:00 loading operations are continuously performed without

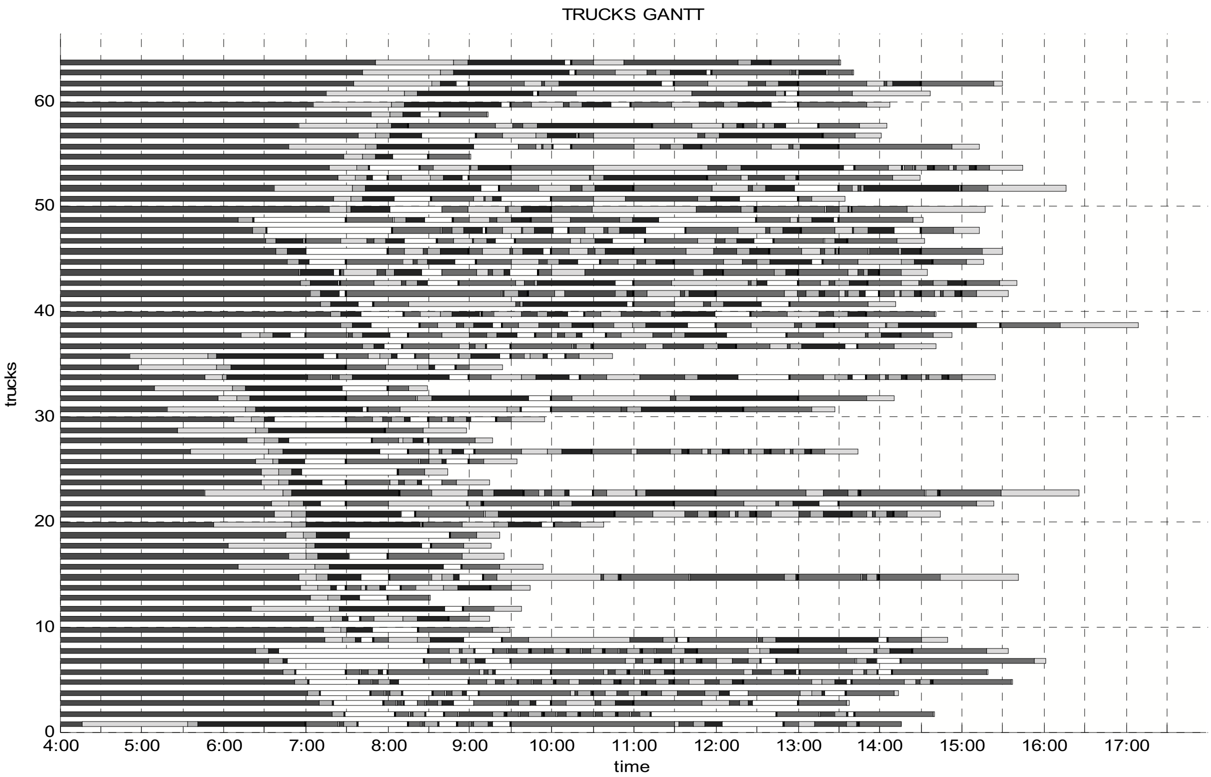


Figure 11: Trucks Gantt diagram (49 vehicles from the internal fleet and 15 hired).

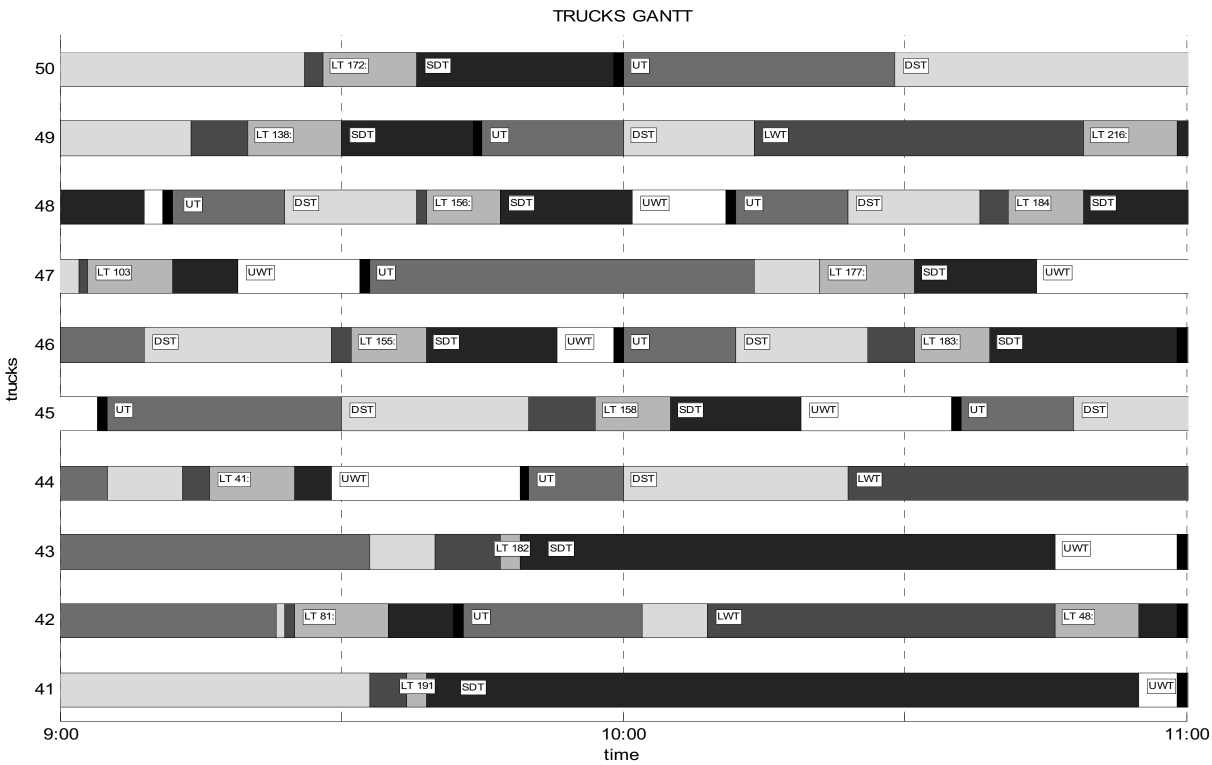


Figure 12: Detail of the trucks Gantt diagram.

pauses. Even the Tilburg–PC3 (the less favorable due to its peripheral position) is not allowed to remain idle in the first part of the day. It can be noted that PC3 has to start mixing earlier than the other ones, due to its larger distance from most of the customers. The concentration of such a large number of deliveries in a relatively short time window implies several noticeable effects. Firstly, many concrete batches have to be mixed considerably earlier than the optimal time, and consequently the trucks delivering these jobs may arrive greatly in advance at the customers' sites. This effect can be noted in Fig. 11 observing the waiting times (depicted in white) before the first task of each truck (Fig. 12 reports a detail of the Gantt chart to better illustrate the sequence of operations scheduled on each truck). Secondly, a great number of trucks is required in this part of the day. In fact, it can be noted in Fig. 11 that many of them are used only until 9:00 AM. After this time, about half of the fleet returns to the relative base location as the supply operations become less critical, as also visible in Fig. 10, where some relatively short idle times between loading operations are allowed in the central part of the morning. In particular, the schedule found by the hybrid GA tends to concentrate these idle times on the Tilburg-PC3, due to the aforementioned distance of this center from most customers sites. As shown in Fig. 9, a second peak of demands occurs at about 1:00 PM, causing four PCs to reenter the uninterrupted loading stages. Finally, after 3:00 PM there are no more demands and the trucks are allowed to return to their base depots.

Let us now consider the performance evaluation over the whole set of considered (hypothetical and real world) problem instances. The 250 different demand patterns have been grouped in five classes of gradually increasing complexity (labelled from *very low*, *low*, *average*, *high*, and *very high* difficulty) based on the average number of hired trucks, and on the average quantity of outsourced production resulting in the solutions found by the four heuristic strategies used for comparisons. These performance indices were chosen to obtain a fairly realistic estimation of the difficulty of the instances, which is not only related to the number and size of the demands, but also strongly affected by the interferences between the various orders, the customer-specified time windows, unloading rates and additional requirements. The result of the experimental investigation is summarized in Table 3. Since in every considered instance the GA-based hybrid approach significantly outperforms all the terms of comparisons, the table reports the average percentile increment of the cost function provided by each heuristic with respect to the average result obtained by running the GA ten times for each instance.

Table 3: Summary of cost increase of the other variants on the one obtained by GA.

Difficulty	Scheduling Policy			
	SD/SIT	SD/LIT	SW/SIT	SW/LIT
Very Easy	14.96%	108.99%	94.91%	174.83%
Easy	43.21%	130.33%	135.79%	215.81%
Normal	44.29%	98.30%	92.24%	142.61%
Hard	49.52%	100.08%	88.89%	132.80%
Very Hard	39.54%	67.85%	55.44%	82.04%

It can be noted that in all the considered cases, the SD/SIT is the best strategy among the methods used for comparison, with an average loss with respect to GA of about 15% in the easiest instances. In fact, when all the PCs mix at a rate that is significantly lower than their maximum capacity, good overall solutions can be easily found by assigning the requests to the nearest PCs. The discrepancy of cost values raises up to nearly 50% in the cases of moderate and high complexity, owing to the optimized distribution of the loads performed by the GA. The reduced difference of costs in the case of very high complexity is due to the fact that in these cases the overall demand exceeds the maximum productive capacity of the supply chain. Thus, also the solutions found by the GA entail a significant amount of outsourced production, with associated additional costs that also flatten the differences between the most effective scheduling policies and the less performing ones.

Finally, we carried out an investigation on the robustness of the found solutions to stochastic perturbations, such as transportation delays due to traffic or other unexpected events. This analysis is obtained with the aid of a discrete-event simulation of a detailed model of the supply chain, developed within the Rockwell Arena 7 discrete event simulation environment. In the simulated scenarios, the actual speed of trucks is modelled with a triangular distribution. In particular, while the median value of the distribution is set equal to the truck speed assigned in the deterministic model, the left- and right-hand half-widths of the distribution are progressively enlarged so as to investigate the effects of transportation delays of increasing size. As mentioned, the tolerance to delays of a given schedule is determined by the amount of truck waiting times LWT and UWT at PCs and at customers' locations, respectively. For instance, if a truck arrives ten minutes late at a delivery location, but it was initially scheduled to reach the

Table 4: Percent number of failed replications for each simulation performed.

MWT	Truck speed distribution - Half width (Km/h)												
(mins)	0.0	2.5	5.0	7.5	10.0	12.5	15.0	17.5	20.0	22.5	25.0	27.5	30.0
5	0%	0%	20%	80%	100%	/	/	/	/	/	/	/	/
10	0%	0%	0%	5%	40%	95%	100%	/	/	/	/	/	/
15	0%	0%	0%	0%	0%	45%	80%	100%	100%	/	/	/	/
20	0%	0%	0%	0%	0%	0%	10%	55%	80%	90%	95%	95%	100%
25	0%	0%	0%	0%	0%	0%	0%	10%	40%	65%	95%	100%	/
30	0%	0%	0%	0%	0%	0%	0%	0%	5%	25%	80%	90%	95%

site fifteen minutes earlier than the customer specified unloading time (i.e. $UWT = 15$ min), the delay is compensated by the waiting time without having any further consequence on the remaining scheduling plan. On the contrary, if a truck returns to a PC later than the scheduled loading time of its next job (i.e. the delay exceeds the planned LWT), the perturbation may determine chained delays that may significantly affect all the successive operations. For this reason, the safety parameter MWT defines the lower bound for all the loading and unloading waiting times. It is worth mentioning that higher values for the MWT will certainly lead to more delay-tolerant solutions, but it will also increase the value of the component of the cost function associated to waiting times.

The evaluation of tolerance to perturbations is carried on a reference instance selected from the class of *hard* difficulty. We consider 12 discrete increments for the half-width of truck speed distribution (up to 30 km/h, which would cause on some long routes delays of more than two hours), and 6 increasing values for the safety parameter MWT. For each combination of half-width and MWT, 20 different replications of the discrete-event simulation of the supply chain are run. The final results are summarized in Table 4, which reports the percentile number of replications in which at least one of the following unacceptable events occurred.

- A truck arrives late at a customer site, either violating the continuity of the unloading process or exceeding the concrete setting time.

- A truck returns late to the PC for its next job, delaying the next loading operations at the PC.

Table 4 shows that solutions capable to fully tolerate even reasonably high variations of average truck speed can be found by appropriately setting the value of the safety parameter MWT. The costs associated to solutions with increasing values of the MWT are summarized

Table 5: Total costs and number of hired trucks relative to the introduction of the safety variables.

MWT (mins)	total cost	hired trucks
0	242610	4
5	350370	11
10	460300	17
15	569485	22
20	632585	25
25	713345	29
30	796315	33

in Table 5. It should be noted that the extension of the safety margin does not only affect the cost associated to waiting times, but obviously also involves cost related to external truck hiring, since the utilization of the internal fleet is significantly reduced by the increased waiting times. The results in the Table 5 indicate that both the overall costs and the number of hired trucks have an approximately linear growth with the safety factor MWT. This particular feature of the proposed model makes it possible to easily determine in advance the value of MWT that provides the desired tradeoff between delay tolerance and final cost associated to the solution found with the proposed GA-based scheduling strategy.

6 Conclusions

In this work, we considered the problem of finding an optimized schedule for the just-in-time production and delivery of ready-made concrete on a set of distributed and coordinated production centers. Our attention was firstly focused on the development of a complete and detailed deterministic model of the considered supply chain, enlightening all the peculiarities that make it considerably different from other formulations of similar scheduling and routing problems. In a subsequent step, we described an effective scheduling algorithm based on the proposed model. 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 requests. The proposed scheduling algorithm was compared with other four constructive heuristics on an industrial case study using a comprehensive set of problem instances. The results obtained illustrate the interesting potential of the proposed approach. Firstly, in the solutions found by the GA the amount of requests that are redirected to external companies, or that need 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. With the aid of a discrete-event simulation campaign, we have shown that schedules capable of tolerating considerable variations of truck average speed can be found with the proposed algorithm.

Our research work is rich of promising directions deserving further investigations. Firstly, even if the proposed GA is able to find satisfactory solutions in short execution times, such an optimization algorithm can be refined in a number of different ways, e.g. devising more efficient crossover and mutation operators. Moreover, a multi-objective version of the proposed algorithm, which is capable of finding the Pareto front of nondominated solutions with respect to the single components of the cost function (distances, cost of outsourcing, waiting times/safey margins) is currently under development. Long term research also involves the investigation of innovative paradigms based on distributed optimization, in which enhanced reactivity and fault tolerances are achieved by distributing the scheduling task between various decision nodes located at each PC of the supply chain, instead of concentrating it to a centralized optimization engine.

References

[1] C. W. Feng, T. M. Cheng, and H. T. Wu. Optimizing the schedule of dispatching RMC trucks through genetic algorithms. *Automation in Construction*, 13(3):327–340, May 2004.

[2] S. Funda Sivrikaya- and G. Ulusoy. Parallel machine scheduling with earliness and tardiness penalties. *Computers & Operations Research*, 26(8):773–787, July 1999.

[3] J. M. 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 *Proceedings of the* 9^{th} *International Conference on Neural Information Processing (ICONIP'02)*, volume 3, pages 1153–1158, Nov. 2002.

[4] 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.

[5] G. Laporte, M. Gendreau, J.-Y. Potvin, and F. Semet. Classical and modern heuristics for the vehicle routing problem. *International Transactions in Operational Research*, 7(4–5):285–300, Sept. 2000.

[6] C. Y. Lee and J. Y. Choi. A genetic algorithm for job sequencing problems with distinct due dates and general early–tardy penalty weights. *Computers & Operations Research*, 22(8):857–869, Oct. 1995.

[7] H. L. Lee, C. T. Tan, K. Ou, and Y. H. Chew. Vehicle capacity planning system: a case study on vehicle routing problem with time windows. *IEEE Transactions on Systems, Man and Cybernetics, Part A*, 33(2):169–178, Mar. 2003.

[8] K. M. Lee, T. Yamakawa, and K.-M. Lee. A genetic algorithm for general machine scheduling problems. In *Proceedings of the Second International Conference on Knowledge-Based Intelligent Electronic Systems*, volume 2, pages 60–66, Adelaide, Australia, Apr. 1998.

[9] Y. Marinakis and A. Migdalas. Annotated bibliography in vehicle routing. *Operational Research – An International Journal*, 2:32–46, 2003.

[10] 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.

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

[12] L. Min and W. Cheng. A genetic algorithm for minimizing the makespan in the case of scheduling identical parallel machines. *Artificial Intelligence in Engineering*, 13(4):399–403, Oct. 1999.

[13] 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.

[14] L. Özdamar. A genetic algorithm approach to a general category project scheduling problem. *IEEE Transactions on Systems, Man and Cybernetics, Part C*, 29(1):44–59, 1999.

[15] 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.

[16] P. Toth and D. Vigo. Models, relaxations and exact approaches for the capacitated vehicle routing problem. *Discrete Applied Mathematics*, 123(1–3):487–512, Nov. 2002.

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

[18] F. Zhang, Y. F. Zhang, and A. Y. C. Nee. Using genetic algorithms in process planning for job shop machining. *IEEE Transactions on Evolutionary Computation*, 1(4):278–289, 1997.

A Definition of Symbols and Acronyms

A.1 Demand related

$r \in \{1, \ldots, R\}$	*Customer* or *demand-related* index. R is the number of customers/requests processed in the considered time horizon.
Q_r	Quantity of cement requested in r-th demand.
$[EDT_r, LDT_r]$	Customer-specified earliest and latest delivery time for request r.
Per_r	User-specified percentage of truck capacity that should not be used.
Mds_r	Maximum size of a delivery allowed by customer r to a single truck.
Fix_r	User-specified fixed waiting time at the destination.
UR_r	Rate of unloading of customer r.
$Tset_r$	Setting time of the type of concrete requested by customer r.

Z_r Number of sub-deliveries in which a request exceeding truck capacity is divided. A sub-delivery is referred to as a job. We assume that all the sub-deliveries to the same customer have the same size. In particular, when a request r can be handled by a single truck, Z_r is equal to 1, otherwise it is computed taking into account the user-specified limitations on the maximum size of delivery Mds_r as follows:

$$Z_r = \left\lceil \frac{Q_r}{\min\{C_{max}\,(1 - Per_r), Mds_r\}} \right\rceil, \tag{39}$$

where C_{max} is the capacity of a single truck, and $\lceil \bullet \rceil$ indicates the *ceiling operation* (rounding off to the nearest larger integer).

A.2 Sub-demand (job) related

$i \in \{1, \ldots, N\}$ *Job-related index relative to the job.* N is total number of jobs to perform. Clearly, it holds:

$$N = \sum_{r=1}^{R} Z_r. \tag{40}$$

The sub-demands are arranged in sequential orders, so that the index i can be interpreted as follows:

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

f_r First job of request r.

l_r Last job of request r. According to this notation, there is a biunivocal correspondence between r and i. For brevity, we denote with r_i the demand to which job i belongs.

LT_i Job loading time.

SDT_i Source to destination travelling time for job i.

DST_{ij} Travelling time between the destination of job i and the source of job j.

SLT_i Earliest loading start time that guarantees the completion of the supply (end of unloading) before the concrete sets.

LLT_i — Latest loading start time that guarantees the completion of the overall delivery within the expiration of the customer-specified time LDT_r.

ELT_i — Latest loading end time that guarantees the completion of the overall delivery within the expiration of the customer-specified time LDT_r; clearly, it holds that $ELT_i = LLT_i + LT_i$.

UWT_i — Waiting time before starting to unload at destination of job i.

UT_i — Job unloading time. As we assume that all the jobs composing a single demand have equal size, the unloading time depends only on the size of the demand Q_r and on the unloading rate of the customer UR_r:

$$\text{if} \quad f_r \leq i \leq l_r \qquad UT_i = \frac{Q_r}{Z_r} \frac{1}{UR_r}. \tag{41}$$

A.3 Depot related

$d \in \{1, \ldots, D\}$ — *Depot-related index.* D is the number of depots.

$\Lambda(\alpha, \beta)$ — Distance between two known locations α and β *(either a depot or a customer site).*

LR_d — Loading rate at depot d.

FLT_d — Fixed loading time at depot d.

Γ_d — Subset of jobs that have source in the depot d.

A.4 Truck related

$k \in \{1, \ldots, K\}$ — *Truck-related index.* K is the total number of trucks. $K = K_c + K_o$, where K_c is the number of trucks of the company, and K_o is the number of additionally hired trucks.

C_{max} — Maximum capacity of a truck.

V — Average speed of the trucks.

$STWD_k$ — Starting time of the working day for truck k.

$ENDWD_k$ — Ending time of the working day for truck k.

DP_k Base location (depot) of truck k, $DP_k \in \{1, \ldots, D\} \cup \{H\}$, where H is the base location for hired trucks.

DF_k Depot where the truck k has to start its first load. This variable is introduced because some trucks (e.g. all the hired trucks), may need to move to a depot different from their base location to pick up their first load.

DL_k Last customer served by truck k before its return to base location.

T_k^{start} Time of departure of the truck k from its base location.

A.5 Task related

$m \in \{1, \ldots, M_k\}$ *Task-related index.* A task of a truck is the delivery of a job to its destination. M_k is the maximum number of tasks allowed to a single truck k.

$$M_k \leq M = \left\lceil \frac{\text{length of the working day}}{\text{minimal length of a task}} \right\rceil . \tag{42}$$

LWT_{km} Waiting time for loading the m-th task of k-th truck.

A.6 Decision variables

$X_{ikm} \in \{0, 1\}$ If the job i is *assigned to* truck k as m-th task, $X_{ikm} = 1$, otherwise $X_{ikm} = 0$.

$Y_{id} \in \{0, 1\}$ If job i is produced at the depot d, $Y_{id} = 1$ and $i \in \Gamma_d \subseteq \{1, \ldots, R\}$, otherwise $Y_{id} = 0$.

$Y_{oi} \in \{0, 1\}$ If the production of job i is outsourced, $Y_{oi} = 1$, otherwise $Y_{oi} = 0$.

A.7 Cost parameters

CP Cost for each minute of travel of a single truck (independently of truck type, or travel condition (loaded, empty)).

PT Loss per m^3 of outsourced product.

CA Penalty for waiting time.

HC Cost per day of an hired truck.

A.8 Safety parameters

MWT Minimal waiting time for a truck before

1. loading its next job at a PC, or
2. unloading the concrete at the customers site.

This parameter is used as safety margin to tolerate transportation delays

Publications in the Report Series Research* in Management

ERIM Research Program: “Business Processes, Logistics and Information Systems”

2004

Smart Pricing: Linking Pricing Decisions with Operational Insights
Moritz Fleischmann, Joseph M. Hall and David F. Pyke
ERS-2004-001-LIS
http://hdl.handle.net/1765/1114

Mobile operators as banks or vice-versa? and: the challenges of Mobile channels for banks
L-F Pau
ERS-2004-015-LIS
http://hdl.handle.net/1765/1163

Simulation-based solution of stochastic mathematical programs with complementarity constraints: Sample-path analysis
S. Ilker Birbil, Gül Gürkan and Ovidiu Listeş
ERS-2004-016-LIS
http://hdl.handle.net/1765/1164

Combining economic and social goals in the design of production systems by using ergonomics standards
Jan Dul, Henk de Vries, Sandra Verschoof, Wietske Eveleens and Albert Feilzer
ERS-2004-020-LIS
http://hdl.handle.net/1765/1200

Factory Gate Pricing: An Analysis of the Dutch Retail Distribution
H.M. le Blanc, F. Cruijssen, H.A. Fleuren, M.B.M. de Koster
ERS-2004-023-LIS
http://hdl.handle.net/1765/1443

A Review Of Design And Control Of Automated Guided Vehicle Systems
Tuan Le-Anh and M.B.M. De Koster
ERS-2004-030-LIS
http://hdl.handle.net/1765/1323

Online Dispatching Rules For Vehicle-Based Internal Transport Systems
Tuan Le-Anh and M.B.M. De Koster
ERS-2004-031-LIS
http://hdl.handle.net/1765/1324

Generalized Fractional Programming With User Interaction
S.I. Birbil, J.B.G. Frenk and S. Zhang
ERS-2004-033-LIS
http://hdl.handle.net/1765/1325

---

* A complete overview of the ERIM Report Series Research in Management:
https://ep.eur.nl/handle/1765/1

ERIM Research Programs:
LIS Business Processes, Logistics and Information Systems
ORG Organizing for Performance
MKT Marketing
F&A Finance and Accounting
STR Strategy and Entrepreneurship

Learning Opportunities And Learning Behaviours Of Small Business Starters: Relations With Goal *Achievement, Skill Development, And Satisfaction*
Marco van Gelderen, Lidewey van der Sluis & Paul Jansen
ERS-2004-037-ORG
http://hdl.handle.net/1765/1429

Meta-heuristics for dynamic lot sizing: A review and comparison of solution approaches
Raf Jans and Zeger Degraeve
ERS-2004-042-LIS
http://hdl.handle.net/1765/1336

A Multi-Item Inventory Model With Joint Setup And Concave Production Costs
Z.P. Bayındır, S.I. Birbil and J.B.G. Frenk
ERS-2004-044-LIS
http://hdl.handle.net/1765/1535

The Level Set Method Of Joó And Its Use In Minimax Theory
J.B.G. Frenk and G. Kassay
ERS-2004-045-LIS
http://hdl.handle.net/1765/1537

Reinventing Crew Scheduling At Netherlands Railways
Erwin Abbink, Matteo Fischetti, Leo Kroon, Gerrit Timmer And Michiel Vromans
ERS-2004-046-LIS
http://hdl.handle.net/1765/1427

Intense Collaboration In Globally Distributed Teams: Evolving Patterns Of Dependencies And Coordination
Kuldeep Kumar, Paul C. van Fenema and Mary Ann Von Glinow
ERS-2004-052-LIS
http://hdl.handle.net/1765/1446

The Value Of Information In Reverse Logistics
Michael E. Ketzenberg, Erwin van der Laan and Ruud H. Teunter
ERS-2004-053-LIS
http://hdl.handle.net/1765/1447

Cargo Revenue Management: Bid-Prices For A 0-1 Multi Knapsack Problem
Kevin Pak and Rommert Dekker
ERS-2004-055-LIS
http://hdl.handle.net/1765/1449

Real-Time Scheduling Approaches For Vehicle-Based Internal Transport Systems
Tuan Le-Anh and M.B.M. De Koster
ERS-2004-056-LIS
http://hdl.handle.net/1765/1452

Individual Telecommunications Tariffs in Chinese Communities: History as a Mirror of the Future, and Relevance for Mobile Service Development in China
H.Chen; L-F Pau
ERS-2004-057-LIS
http://hdl.handle.net/1765/1582

Activating Knowledge Through Electronic Collaboration: Vanquishing The Knowledge Paradox
S. Qureshi and P. Keen
ERS-2004-058-LIS
http://hdl.handle.net/1765/1473

A Grounded Theory Analysis Of E-Collaboration Effects For Distributed Project Management
S. Qureshi, M. Liu and D. Vogel
ERS-2004-059-LIS
http://hdl.handle.net/1765/1448

Collaborative Infrastructures For Mobilizing Intellectual Resources: Assessing Intellectual Bandwidth In A Knowledge Intensive Organization
R. Verhoef and S. Qureshi
ERS-2004-060-LIS
http://hdl.handle.net/1765/1474

A Phenomenological Exploration Of Adaptation In A Polycontextual Work Environment
P.C. van Fenema and S. Qureshi
ERS-2004-061-LIS
http://hdl.handle.net/1765/1612

Satisfaction Attainment Theory As A Model For Value Creation
R.O. Briggs, S. Qureshi and B. Reining
ERS-2004-062-LIS
http://hdl.handle.net/1765/1450

Diagnosis In The Olap Context
Emiel Caron, Hennie Daniels
ERS-2004-063-LIS
http://hdl.handle.net/1765/1492

A Deterministic Inventory/Production Model With General Inventory Cost Rate Function And Concave Production Costs
Z.P. Bayındır, S.I. Birbil and J.B.G. Frenk
ERS-2004-064-LIS
http://hdl.handle.net/1765/1536

On And Off The Beaten Path: How Individuals Broker Knowledge Through Formal And Informal Networks
Rick Aalbers, Wilfred Dolfsma & Otto Koppius
ERS-2004-066-LIS/ORG
http://hdl.handle.net/1765/1549

Fractional Programming
J.B.G. Frenk And S. Schaible
ERS-2004-074-LIS
http://hdl.handle.net/1765/1610

Introduction To Convex And Quasiconvex Analysis
J.B.G. Frenk and G. Kassay
ERS-2004-075-LIS
http://hdl.handle.net/1765/1611

Simple heuristics for push and pull remanufacturing policies
Erwin A. van der Laan and Ruud H. Teunter
ERS-2004-090-LIS
http://hdl.handle.net/1765/1786

Genetic Algorithms In Supply Chain Scheduling Of Ready-Mixed Concrete
D. Naso, M. Surico, B. Turchiano and U. Kaymak
ERS-2004-096-LIS