MULTI-OBJECTIVE EVOLUTIONARY SCHEDULING OF DISTRIBUTED SUPPLY NETWORKS

David Naso and Michele Surico

Dipartimento di Elettrotecnica ed Elettronica
Politecnico di Bari – Via Re David, 200 – 70125 Bari – Italy
Fax: +39 - 080 5460 410 – *e.mail: naso@poliba.it*

Abstract: This paper considers the problem of scheduling a distributed network of production centers supplying a quickly perishable good that has to be produced just-in-time and delivered within customer-specified strict time windows. The problem includes several planning, scheduling and routing problem, each notoriously affected by nearly prohibitive combinatorial complexity. Ideal solutions should provide a good compromise between production costs (resource utilization, delivery costs) and tolerance to stochastic perturbations (transport delays). We propose a novel multi-objective meta-heuristic approach based on a hybrid genetic algorithm combined with constructive heuristics. The algorithm is designed to return a set of solutions with different cost and risk tradeoffs. The effectiveness of the approach is confirmed by a comparison with other recently proposed methods.

Keywords: Supply Networks, Scheduling, Meta-Heuristics, Multi-Objective Genetic Algorithms.

1. INTRODUCTION

Supply networks (SN's) can be viewed as organizations of partially autonomous production and distribution centers through which goods are processed and delivered to customers. The global operation of such complex networks involves the scheduling and the coordination of many activities within and across centers for fast and timely product delivery and low inventory (Luh et al., 2003). Effective coordination policies are particularly important for "just-in-time" SN's because their flow of material is triggered by dynamic orders with strict delivery time requirements, while activities are strongly interrelated by technical constraints, so that a delay of one process may have domino effects on the other activities linked through precedence relationships or through sharing of common resources. From the mathematical viewpoint, the organization of SN activities 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 each independent from the other ones, the mentioned logistic problems suffer from a nearly prohibitive combinatorial complexity. On the other hand, there is a strong need for approaches that are capable of finding satisfactory solutions in short computation times, to cope with the unpredictable dynamic changes (new orders, delays, failures) that often impose a sort of continual revision of the planned solution.

This paper considers a SN for the production and delivery of a quickly perishable good (ready-mixed concrete, RMC), which has to be produced on-demand and delivered within strict time-windows. The SN consists of a network of independent production centers (PC's) serving a set of distributed customers. Some PC's host a fleet of trucks to deliver the RMC, but a few ones do not own carriers, and explicitly rely on the other PC's for transportation. Large demands must be transported by several trucks, which have to be properly synchronized because the unload process at the customer site must be continuous (there cannot be significant pauses between the completion of a truck unload and the start of the next one). Individual PC goals are multifaceted. Each PC aims at organizing its activities so to associate high resource utilization to low transportation costs and timeliness of the deliveries, the latter being particularly crucial for the characteristics of the supplied good. The problem is extremely complex, not only for the typical combinatorial complexity that is particularly prohibitive in such large scale systems, but also for the high number of constraints deriving from the perishable nature of RMC, and for the conflicting nature of the cost and timeliness objectives.

At present, many companies tend either to rely on skilled operators that work out production plans basing on their experience (Matsatsinis, 2004, Feng, 2004) or to plan production operations on short time horizons, sacrificing the optimization on longer term to reduce the risk of delayed delivery (Tommelein, 1999). To overcome the limitations of this practice, this paper proposes a multi-objective approach combining search methods from the area of Evolutionary Computation (EC) with constructive heuristics developed for the considered case. The proposed method integrates the following tools:

- A detailed model of the logistic problem that unambiguously specifies the decision variables of the problem, the technical requirements, and the constraints that must be fulfilled in practice.
- A set of constructive heuristics that are able to process any initial assignment of the decision variables until a legal solution (i.e. satisfying all the constraints) is obtained.
- A Multi-Objective Genetic Algorithm (MOGA) that seeks for the Pareto-front of solutions with respect to both cost- and risk-related objectives.

The proposed approach is based on an industrial case-study, employing data provided by a network of suppliers operating in the northern part of the EU.

2. PROBLEM AND LITERATURE OVERVIEW

We consider a network of D depots or PC's ($d\in\{1,...,D\}$ is the depot index) located in a given geographical area, which receive and process a set of R requests or orders from different customers ($r\in\{1,...,R\}$ is request index). An order r consists of a customer-specified delivery time window [EDT_r, LDT_r] (earliest and latest delivery time), a required amount Q_r, and a delivery location. The SN is also equipped with a fleet of K trucks ($k\in\{1,...,K\}$) to deliver the product to customers. If a demand exceeds the capacity of a single truck, it is split in a number of sub-demands or jobs ($i\in\{1,...,N\}$ is the job index, and N is the total number of jobs of the SN), which will be delivered to customers one after the other. In case none of the nodes of the SN is able to service a certain (fraction of) demand, the production of the exceeding amount can be outsourced to external suppliers with a consequent loss of profit. When none of the trucks from the internal fleet is available for the delivery of a given order, a truck from an external company can be hired with additional costs.

A detailed mathematical model has been developed to obtain a comprehensive formal description of the problem that could be used by automatic search techniques. For brevity, the full mathematical model is omitted here (most of its detail can be found in (Naso et al., 2004)), while the reminder of this section overviews some key elements. Let us define a *task* of a truck as the delivery of a job to its destination, and introduce the task index $m\in\{1,..., M_k\}$, where M_k is the last task of truck k. We consider three types of binary 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 otherwise Y_{id} = 0.

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

Further nomenclature about the operations of each truck is introduced in Fig.1, which describes the sequence of operation of a truck. All the time intervals and parameters in the model are assumed deterministic and known a priori.

The SN scheduling must take into account several types of constraints, related to logic assignment of decision variables (e.g. a job can be assigned only once to one truck), to overlap prevention (e.g. a (un)load operation at a PC can only start when the previous one is ended), to RMC lifetime (the unload

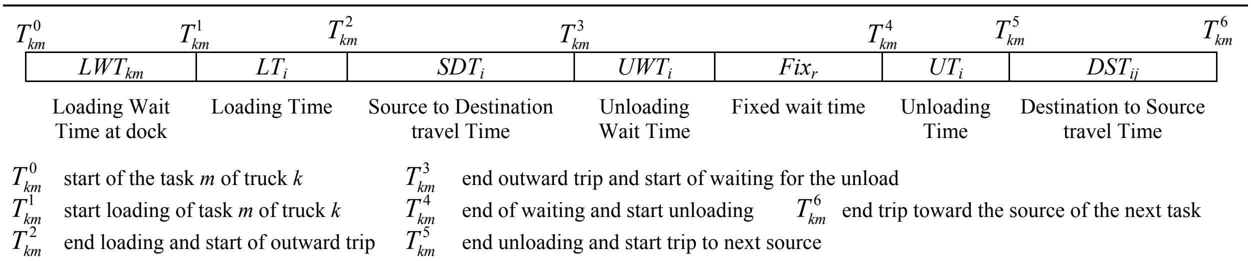

Fig. 1. Sequences of operations for the single task

must be completed before the RMC starts to set). A peculiar constraint that makes delays particularly dangerous in terms of domino effects is related to the continuity of unloading operations:

$$\sum_{k=1}^{K}\sum_{m=1}^{M} X_{ikm}\cdot T_{km}^{5} = \sum_{k=1}^{K}\sum_{m=1}^{M} X_{(i+1)km}\cdot T_{km}^{4} \qquad (1)$$

This condition must be fulfilled by all the jobs *i* originating from a demand split. The waiting times, indicated in Fig.1 as LWT_{km} and UWT_i, are other key-variables of the model. They represent the main idle times before (un)loading. Tight schedules with short waiting times imply better resource utilization while longer waiting times make the schedule more delay-tolerant. For this reason, an user-defined lower bound (*minimum waiting time*, MWT) for all the waiting times is introduced in our model, and additional constraints are added so to ensure that all the waiting times are longer than the minimum allowed threshold.

As mentioned, scheduling goals are related to production and delivery costs and timeliness of deliveries. Even assuming deterministic operation and transportation times, simultaneously achieving these two objectives is difficult. Moreover, an ideal schedule must be able to tolerate unpredictable stochastic perturbations (e.g. transportation delays due to traffic). Hereafter, this aspect is referred to as *robustness* of a solution. In general, cost and robustness are conflicting objectives because, as mentioned, tight schedules are also more sensitive to unexpected delays, especially when activities are tightly coupled by synchronization constraints as (1). On the other hand, the insertion of larger time buffers involves a significant cost growth since it reduces the actual resource utilization. Thus, the SN scheduling problem can be viewed as a bi-objective search problem in which ideal solutions are those which guarantee a good tradeoff between low overall costs and a satisfactory robustness to unpredictable delays.

From the mathematical programming viewpoint, the production cost of a solution is the sum of three terms:

$$C = C_{transport} + C_{waiting} + C_{extra} \qquad (2)$$

All the cost components depend on the decision variables and on the consequent schedule of all the activities. The transportation costs account for the distances traveled by all the trucks of the fleet, including the initial and final trips from and to the respective base location. The waiting costs account for all the truck waiting times (*LWT, UWT, FIX*). The extra costs incorporate all the additional costs related to outsourced production, hired trucks, and overtime work for truck drivers. The schedule robustness is estimated with the following risk index:

$$RF = 1 - \frac{Q}{Max(Delay)} \qquad (3)$$

where

$$Q = \alpha \cdot avg(WT_i) - \beta \cdot std(WT_i) \qquad (4)$$

In (4) WT_i is the sum of the waiting times associated to job *i* (*UWT, LWT, FIX*), α and β are two weighting factors, and *avg* and *std* are average and standard deviation respectively. The index Q aims at evaluating the way time buffers are distributed in the schedule. Ideally, waiting times should be sufficiently long and evenly distributed across the whole activity schedule. Thus, their average should be maximized and the standard deviation minimized. In (3), *Max(Delay)* is the maximal expected delay of a travel, which can be easily set by plant managers according to historical data.

It can be observed that the global SN scheduling includes several logistic problems that have been extensively considered in past literature, such as the vehicle routing problem (VRP, Marinakis and Migdaleas, 2003), scheduling of parallel machines (Serifoglu and Ulusoy, 1999), scheduling with earliness/tardiness penalty (Ventura and S. Radhakrishnan, 2003). However, the SN size, and also the type and number of constraints, make difficult to directly apply and integrate the single approaches. An introductory overview of RMC delivery is available in (Tommelein, 1999). Similarly (Matsatsinis, 2004), surveys the main peculiarities of RMC SN's, and focuses on vehicles (trucks and pumps) routing. Recent researches also propose approaches based on Genetic Algorithms (GA, Garcia et al. 2002, Feng et al. 2004). Note that these works consider only small-size instances, a single depot and an unlimited fleet of vehicles, while the proposed approach is devised and validated on problems with the same size and complexity of a real-world large scale SN.

3. THE HYBRID MULTI-OBJECTIVE GENETIC APPROACH

We search for an effective trade-off between costs and robustness using a MOGA. Such an algorithm is able to simultaneously take into account two or more objectives, and find a set of solutions which approximate the Pareto front (Deb et al. 2002). MOGA's are in general more complex and computation-demanding than normal GA's, because they must perform a higher number of comparisons to rank individuals, and because they need specific mechanisms to prevent the concentration of the search on excessively narrow segments of the Pareto front (Deb et al. 2002). Due to problem size and complexity, using a GA to optimize all the free variables of the problem would involve an unsustainable computational cost. Thus, the GA is used in conjunction with fast local heuristics that permit to reach optimized solutions with short execution times (few minutes on a Pentium IV PC platform) even for the largest instances derived from the available data. In particular, the GA constitutes the main search engines, which assigns demands to depots (decision variables Y_{id}) and defines the order of priority by which the demands are scheduled for production. At every time a new solution has to be evaluated, the GA launches a Constructive Heuristic Algorithm (CHA), which starts from the assignment given by the GA and (1) builds schedules satisfying all the described constraints, (2) assigns the non-outsourced jobs to the trucks. The overall structure of the search algorithm in a hi-level generic pseudo-code is reported in Fig.2. For brevity, the next subsections give a short overview of the basic mechanisms underlying the proposed search algorithm.

3.1. The MOGA

The MOGA is an adapted version of the improved Non-Dominated Sorting GA (NSGA-II, (Deb et al. 2003), an effective algorithm widely referred in technical literature. The NSGA-II is devised to explore continuous domain, and most of its functions have to be adapted to deal with our combinatorial problem. A first important issue regards the solution encoding. In our approach, each chromosome is made up of two separate parts, both containing *R* elements, as described in Fig.2 for *R*=6.

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

Fig. 2. a single chromosome

The first part defines the initial values for the decision variables Y_{id} that assign the demands to the depots. At this stage of the decision process, it is assumed that all the jobs composing a split request are produced at the same PC. For instance, gene r_2 indicates that the request 2 is assigned to PC 3. The second part of the chromosome establishes the order in which the *R* requests are considered when building the schedule for the production chain (e.g. request r_4 -scheduled on PC 1- is allocated first, followed by r_5 -on PC 2- and so on). The values of all decision variables not assigned in the chromosome are computed later on by the CHA. This inner module is in charge of constructing a legal solution starting from the partial assignment of decision variables specified by the chromosome. New crossover and mutation operators have been designed for the specific encoding schema. Both operators randomly select a point in the chromosome, and apply a different operator if the selected point is in the first or in the second part of the string, always producing legal offspring solution. Technical details on these operators can be found in (Naso et al. 2004). When the (first) new population of solution is completed, the CHA is run to construct a feasible schedule from each chromosome. Subsequently, the two fitness functions, the cost C and the risk index RF, are computed for each individual. Then, the selection of the best solutions for reproduction in the next population is carried out with the same hybrid ranking/crowding based method of NSGA-II, whose details can be found in (Deb et al. 2002).

3.2. The Constructive Heuristic Algorithm

The CHA is composed of two main modules. The first one (the Depot Schedule Builder or DSB) is in charge of building the schedule of the production for all the jobs, and the second one (the Truck Schedule Builder or TSB) deals with the organization of the transport operations, i.e. it assigns jobs and routes to trucks. It is important to remark that these modules are sequential, i.e. the TSB cannot modify the PC schedule built by the DSB. In principle, this decomposition may lead to sub-optimal solution even though, in spite of this theoretical limitation, our approach always provides solutions that outperform the reference strategy.

The DSB processes requests following the order of priority specified by the chromosome. For each demand, the DSB makes a preliminary set of feasibility checks (e.g. if the distance between the assigned PC and the customer permits the end of the unloading before the RMC setting time). If some constraints are violated, the assignment is modified so to overcome the cause of the violation. Then the DSB attempts to place the start of the loading window of the first job so that the *SDT* ends exactly *MWT* + *FIX* before the customer-specified *EDT*, which corresponds to a sort of ideal loading time. If this window overlaps with a previously assigned job, the DSB makes a series of attempts to overcome the overlaps by rearranging the job sequences without violating other constraints. As a result, among other possibilities, it may happen that some jobs are scheduled to be delivered earlier than the ideal unloading time, thus with a larger-than-desired time buffer. If no adjustment guarantees the feasible schedule of job delivery, the DSB marks the job as undeliverable (its scheduling will be resumed later on), and proceeds to place the second (or subsequent) job(s) of the demand to the specified PC. The scheduling of this second job must also consider that enough time must be available before the job start so to allow the completion of the preceding job(s). If also the second job cannot be assigned to the PC, the algorithm tries with the subsequent ones, until either one of the jobs is assigned to the PC, or none of the jobs composing a request can be scheduled on the PC. In the latter case (100% jobs of the request is not scheduled on the PC specified in the chromosome) the chromosome is changed and the procedure re-examines the assignment of the request on other PC's in order of shortest distance from the customer's site. In any case, the DSB tries (if possible) to assign all the jobs of a split request to the same PC, always verifying that the unloading of each job can start exactly when the preceding one is expected to end. After examining all the demands, the DSB attempts to allocate the production of the undeliverable jobs in the idle time intervals of other PC's, starting from the nearest to the customer's site. Several insertion procedures are examined for each undeliverable job. Finally, if none of these successfully places it on a PC, the job has to be outsourced.

Once the DSB ends its task, the TSB processes the truck schedule so to guarantee that a vehicle is available at the expected loading time of each non-outsourced job. The TSB is a nested sequence of various heuristic strategies devised to optimize truck utilization, attempting to simultaneously minimize the traveled distances and the idle times. Basically, jobs are assigned to trucks in order of their T^1_{km}. Trucks are considered in decreasing order of their Available Time (*AT*) defined as the time at which the truck can reach the PC after completing the preceding tasks. The assignment strategy is referred to as *Shortest (truck) Idle Time* (SIT), because it assigns higher preference to the latest truck that can arrive to the PC. In this way, it attempts to better exploit the truck utilization, concentrating as much as possible the jobs on some trucks while leaving some other vehicles with longer idle times. The latter ones can be profitably exploited to serve PC's that are not equipped with an internal fleet. The trucks with the same *AT* are sorted in increasing order of the distance from the source PC, in order to account for distance-related cost criteria. The job is finally assigned to the truck by guaranteeing it will be available at least *MWT* before the scheduled job loading start. If no truck is available to fulfill this requirement a request for a hired truck is issued for the deliver of the job.

4. THE CASE STUDY

Our research investigation focuses on experimental data observed during several working days of a SN with 5 autonomous PC's located in the northern EU. The fleet of trucks consists of 49 vehicles housed in two PC's, and the customers are spread over the area surrounding the network of suppliers. The available data confirms the typical statistical distributions observed in similar cases (Matsatsinis, 2004). In all our instances the weighting factors α and β have been set respectively to 1 and 0.2, so to guarantee that the index *Q* is always positive, while *Max(Delay)* has been set to 90 (min.). We consider two other scheduling algorithms as reference policies for performance comparison. The first one is a constructive procedure that incorporates the typical decision criteria used by expert plant schedulers. Basically, this algorithm (hereafter SD-SIT) attempts to assigns the requests to the PC closest to customer's site (*Shorter Distance*, SD), and the jobs to the trucks with the previously introduced SIT criterion. The second reference algorithm is a single-objective GA (SOGA) recently proposed in (Naso et al. 2004). This GA is designed to address only cost minimization, and uses a different version of the CHA. The two algorithms were recently compared with several other heuristic strategies (Naso et al. 2004) on problem instances of differing complexity. The SD-SIT was in all the considered cases the best non-evolutionary approach, while the SOGA was always able to outperform all the other methods.

We consider a reference instance describing a typical busy working day of the SN, with 71 requests split in more than 300 jobs. The main configuration and cost parameters used in the decision problem are summarized in Table I. All the considered algorithms are run multiple times, with gradually increasing values of the MWT. In this way, different solutions are found with increasing costs and robustness to delays. Differently from SD-SIT and SOGA, each returning only one solution, the MOGA provides a Pareto-front of non-dominated solutions describing different tradeoffs between cost and robustness. For an immediate comparison, Table II and III report the values of the cost and risk objectives of the two solutions in the extremes of the front (those with the minimum cost and minimum risk, respectively), while Fig.3 provides a graphical comparison using the two objectives as Cartesian axes for the case of MWT=15mins.

Table I: cost parameters and configuration of the algorithm

cost for each Km of travel of the trucks	10
penalty for idle time	15
loss of income for m^3 of concrete to outsource	2000
cost of an hired truck	10000
extra pay for truck drivers' overtime minute	5
population size (randomly generated)	100
termination condition (calls to CHA)	2500
crossover rate	33%
mutation rate	33%

Table II: risk function for the case study

Algorithm \ MWT	SD/SIT	SOGA	MOGA high cost	MOGA low cost
5	0.856	0.877	0.799	0.862
10	0.795	0.823	0.739	0.813
15	0.751	0.769	0.695	0.762
20	0.691	0.724	0.638	0.691
25	0.634	0.649	0.597	0.644
30	0.584	0.606	0.546	0.601

Table III: cost function for the case study

Algorithm \ MWT	SD/SIT	SOGA	MOGA high cost	MOGA low cost
5	492301	432665	686512	457210
10	612462	550178	786999	545475
15	690891	609210	860848	627810
20	809216	706759	938815	728630
25	905416	811440	963676	826405
30	981239	896618	1081105	897000

As in the case reported in Fig.3, the solutions found by SD-SIT are always dominated by those of the

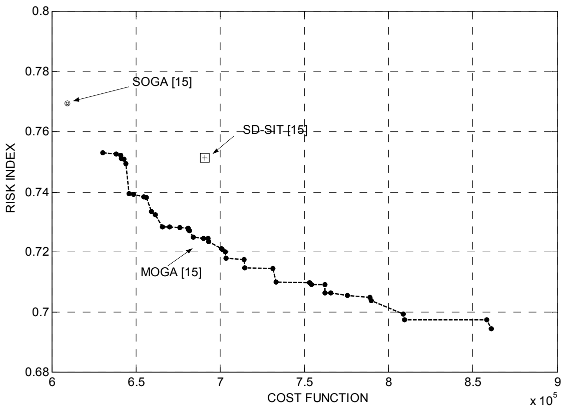

Fig. 3. A comparative analysis of the algorithms

MOGA. In some cases, the SD-SIT yields solutions that have the same overall cost of those found by the MOGA with shorter values of MWT, a circumstance that indicates that significantly better solutions can be obtained by using the proposed multi-objective approach. For what concerns the SOGA, it is able to provide solutions whose costs are sometimes lower than those found in the front of the MOGA (for a given value of the MWT, the SOGA solution often lies near the upper-left side of the front, as in Fig.3). These solutions have in general a high value of the RF, and thus are potentially useful only for cases in which it can be guaranteed that any delay will never exceed the given value of the MWT. To obtain a further validation of the actual performance offered by the considered policies, a discrete-event simulation model of the SN has been developed, in which truck routes are subject to stochastic delays of variable distribution. The investigation confirmed that the probability of critical events (e.g. the loss of RMC loads for excessive delay, the discontinuous unload of large demands, etc.) is significantly reduced in the solutions with low risk index.

5. CONCLUSIONS

We have considered a novel meta-heuristic approach based on a multi-objective evolutionary algorithm combined with constructive heuristics for addressing production and delivery with time constraints on both the earliness and the lateness of supply. Our experimental investigation shows that such a hybrid approach is able to provide an effective scheduling algorithm based on a detailed model of the network. The comparison of the multi-objective method with other heuristic algorithms has confirmed the potentialities of the proposed method on two counts. Firstly, the user is provided with a set of different schedules, each corresponding to a different ratio of production cost and tolerance to unexpected delays. Secondly, the multi-objective approach is able to obtain a satisfactory tradeoff front in the same execution time of the other single-objective heuristics, which can be made short enough to perform real-time rescheduling in case new orders are received during the working day (the current prototypes of the GA codes are programmed in Matlab language and take about 15 minutes to converge on a PC Pentium based platform). Besides the extension to the case of dynamic rescheduling, the current research is focused on the fusion of DSB and TSB in an integrated procedure, on the development of discrete-event simulation tools for the automatic detection of critical parts of the schedules, and an improved strategic allocation of time buffers of variable size.

REFERENCES

Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation,* **6**, 182 – 197.

Feng, C. W., Cheng, T. M., Wu, H. T., 2004. Optimizing the schedule of dispatching RMC trucks through genetic algorithms. *Automation in Construction*, **13**, 327 – 340.

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

Luh, P. B., Ni, M., Chen, H., Thakur, L. S., 2003. Price-Based Approach for Activity Coordination in a Supply Network. *IEEE Trans. on Robotics and Automation*, **19**, pp. 335-346.

Marinakis, Y., Migdalas, A., 2003. Annotated Bibliography in Vehicle Routing. *Operational Research–An International Journal*, **2**, pp. 32–46.

Matsatsinis, Nikolaos F., 2004. Towards a decision support system for the ready concrete distribution system: A case of a Greek company. *European Journal of Operational Research*, **152**, 487 – 499.

Naso, D., Surico, M., Turchiano, B., Kaymak, U., 2004. Genetic algorithms for supply-chain scheduling: a case study on ready-mixed concrete, to appear on *European Journal of Operation Research*.

Serifoglu, F. S., Ulusoy, G, 1999. Parallel machine scheduling with earliness and tardiness penalties. *Computers & Operations Research*, **26**, 773 – 787.

Tommelein, I. D., Li, A., 1999. Just-In-Time Concrete Delivery: Mapping Alternatives for Vertical Supply Chain Integration. Proceedings of the Seventh Annual Conference of the International Group for Lean Construction, Berkeley, California, pp. 97 – 108.

Ventura, J. A., Radhakrishnan, S., 2003. Single machine scheduling with symmetric earliness and tardiness penalties. *European Journal of Operational Research*, **144,** 598 - 612.