The Dance of the Thirty-Ton Trucks: Dispatching and Scheduling in a Dynamic Environment

Martin Durbin
Optimization Solutions Group, Decisive Analytics Corporation, Arlington, Virginia 22202,
martin.durbin@dac.us

Karla Hoffman
Department of Systems Engineering and Operations Research, George Mason University, Fairfax, Virginia 22030,
khoffman@gmu.edu

We report on the application of operations research to a very complex scheduling and dispatching problem. Scheduling and dispatching are never easy, but the scheduling of concrete deliveries is particularly difficult for several reasons: (1) concrete is an extremely perishable product—it can solidify in the truck if offloading is delayed by a few hours; (2) customer orders are extremely unpredictable and volatile—orders are often canceled or drastically changed at the last minute; (3) the concrete company overbooks by as much as 20% to compensate for customer unpredictability; (4) many orders require synchronized deliveries by multiple trucks; (5) when a truck arrives at a customer site, the customer may not be ready for the delivery, or a storm may negate the ability to use the concrete; and (6) most of the travel takes place in highly congested urban areas, making travel times highly variable.

To assist the dispatchers, schedulers, and order takers at this company, we designed and implemented a decision-support tool consisting of both planning and execution tools. The modules determine whether new orders should be accepted, when drivers should arrive for work, the real-time assignment of drivers to delivery loads, the dispatching of these drivers to customers and back to plants, and the scheduling of the truck loadings at the plants.

For the real-time dispatching and order-taking decisions, optimization models are solved to within 1% of optimality every five minutes throughout the day. This nearly continuous reoptimization of the entire system allows quick reactions to changes. The modeling foundation is a time-space network with integer side constraints. We describe each of the models and explain how we handle imperfect data. We also detail how we overcome a variety of implementation issues.

The success of this project can be measured, most importantly, by the fact that the tool is being ported by the parent company, Florida Rock, to each of its other ready-mix concrete companies. Second, the corporation is sufficiently convinced of its importance that they have begun promoting this methodology as a “best practice” at the World of Concrete and ConAgg industry conventions.

Subject classifications: vehicle scheduling; real-time concrete truck scheduling; integer programming applications: dispatching; network flows: time-space networks.

Area of review: OR Practice.

History: Received January 2004; revisions received July 2005, October 2005; accepted October 2005.

1. Introduction

The planning, scheduling, dispatching, and delivery of goods and services remain a challenging problem for products with long shelf lives and relatively predictable demands and delivery times. In the concrete industry, the challenge is dramatically increased due to customer requirements for multitruck, time-synchronized deliveries of a perishable product. Additionally, weather and traffic conditions can adversely affect expected travel time in an environment where more than 90% of orders are modified during the day of delivery. The dynamic nature of the problem requires constant schedule revision.

To solve this problem, a decision-support tool was created that consists of both planning and execution modules. These modules assist both customer service representatives (CSRs) and dispatchers in evaluating thousands of possible delivery alternatives, and provide recommendations based on customer demands and current fleet conditions. This paper describes the group of optimization models required to implement this decision-support tool, and details how the tool overcomes the problems of imperfect data. The foundation for the solution is a time-space network representation of the problem that generates multiple delivery alternatives for each customer order. Choosing a single delivery alternative for each customer adds restrictive integrality constraints to the network model.

The solution of the model formulation assists CSRs and dispatchers in determining: (1) the feasibility of accepting additional orders, (2) the arrival times for drivers reporting to work, (3) the scheduling of all orders, (4) the real-time
assignment of drivers to delivery loads, (5) the dispatching of these drivers to customers and back to plants, and (6) the scheduling of truck loading at the plants. These determinations are made by mathematical models incorporating exact optimization techniques.

The standard dispatching method utilized in the concrete industry is truck based. In truck-based dispatching, each order is assigned a specific number of trucks and a specific plant as the source of the product. These trucks then make trip after trip to the same customer until all of the concrete requested by the customer has been delivered (as illustrated on the left side of Figure 1). The static allocation of trucks and plants can be inefficient in a dynamic environment. Consequently, demand-based dispatching, a method for dynamically dispatching trucks, has been developed. In demand-based dispatching, trucks are assigned to a specific delivery (as opposed to an entire job) when they enter the yard of a plant. When the drivers complete deliveries, they are then directed to a plant, which may or may not be the same plant from which they received their previous load of concrete. Dispatching trucks in this intelligent, responsive fashion results in a complex, intertwined movement of trucks throughout the day (as illustrated on the right side of Figure 1). This intricate behavior is called the dance of the thirty-ton trucks.

The demand-dispatching formulation can be thought of as a relaxation of truck-based dispatching formulation, where the restriction to always return to the home plant is relaxed. When thought of in this way, it is clear that the demand-dispatching formulation will create an answer that allows a far greater range of alternatives and is therefore likely to provide lower-cost solutions than the solution of the truck-based formulation. In truth, prior to this project, schedulers had no computer algorithms to assist them, and therefore were unlikely to solve optimally the more restricted truck-based scheduling problem.

Our models incorporate transportation costs, material costs (which can differ substantially across plants), and delivery revenues. Transportation costs are based on expected travel times and time spent at the customer site. Expected travel times are generated from historical point-to-point data collected through the use of global position system (GPS) technology. Each day new data is acquired and the point-to-point estimates are refined. Dispatchers are able to override the estimates as they learn of any traffic jams or other issues that could impact the travel time.

2. The Problem

This research began when a large concrete company, Virginia Concrete, asked for assistance with a queueing problem. Virginia Concrete is an innovative company that was in the process of installing GPS equipment and sensors on board every truck in their delivery fleet. This GPS equipment allows operators to track the position of trucks, and provides essential information regarding the truck statuses. Additionally, each truck is equipped with an on-board computer and mobile digital transmitter to support data transfer and communication between the truck and the central dispatch center.

Virginia Concrete noticed that trucks were consistently queueing up at both their plants and customer sites, and hoped that with the availability of the new GPS-generated fleet data, they would be able to increase the efficiency of their delivery process. After investigating the problem, it became clear that Virginia Concrete had a complex scheduling problem requiring the integration of order taking, planning, and dispatching. The entire process had to be analyzed to preclude backlogs in any one part of the system.

Virginia Concrete delivers 400–600 loads per day with approximately 125 trucks, which results in 100,000–150,000 loads per year. They estimate that saving a modest five minutes on each load will generate between $500,000 and $750,000 in annual savings. Additionally, dispatching is a high-pressure occupation where efficiency is highly dependent upon the skills of individual dispatchers. When a highly effective dispatcher is absent, the operational efficiency of the company is negatively impacted. Similarly, plant scheduling, next-day scheduling, and the determination of driver arrival times are costly and time-consuming processes. Virginia Concrete believed that the development of a decision-support tool would not only generate savings, it would also (1) decrease stress for dispatchers, schedulers, and plant managers; (2) create consistency in these functions across the company; (3) reduce the time required to train new dispatchers; and (4) enable easier substitution and relocation of dispatchers.

Before explaining the solution approach taken, it is necessary to explain the means by which concrete is delivered. There are several steps involved in delivering concrete. First, the truck must be loaded with the particular mix of concrete that the customer has requested. Next, the driver is required to wash the truck, ensuring that no material is on the exterior of the truck. After washing the truck, the

![Figure 1. The dance of the thirty-ton trucks.](image)
driver begins transit to the customer site. Upon arrival at
the customer site, and assuming that the customer is ready
for the arrival of up to 20 tons of concrete, the driver moves
the truck into proper position. This process is called stag-
ing. After the truck is in the proper location, the truck begins pouring the concrete. After completing the pouring
of the concrete, the driver pulls away from the unload site
and again washes the truck. After successfully washing the
truck, the driver begins transit to a plant, although not nec-
essarily to the plant from which the driver started. At the
plant, the driver will either be assigned another delivery or
will be sent home for the day.

There are two decision points during the delivery pro-
cess. The first decision point is the selection of individual
trucks for particular delivery assignments. The second
decision point is the determination of which plant to direct
a truck to after the completion of the delivery. However,
before discussing these decisions, additional complexities
associated with the delivery of concrete will be discussed.
• Concrete is a perishable product and will harden in
the truck after a given amount of time.
• Many concrete orders require more than one truckload
of concrete. In this case, trucks are required to arrive at the
customer site at a specified interarrival rate.
• Concrete must be poured in a continuous fashion.
Hence, once delivering concrete to a customer has begun,
it must be continued until all of the concrete that the cus-
tomer requires has been delivered.
• The customer rarely knows the exact size of the order.
However, even when the customer incorrectly estimates the
amount of concrete needed, the company must supply all
that is required.
• Most trucks hold nine cubic yards of concrete and
weigh approximately thirty tons when loaded. Although
Virginia Concrete has some trucks that can hold 10 cubic
yards, we treat all trucks as interchangeable for this mod-
eling effort. If a 10 cubic yard truck is used, then the sys-
tem updates the remaining concrete needs of the customer
accordingly.
• Orders may be restricted to a subset of the trucks,
drivers, and/or plants.
• Plants are limited in the number of trucks per hour
that can be loaded with concrete.
• All concrete is not the same. Therefore, once a truck
is loaded with concrete, there is little chance of diverting
the delivery to another customer.
• The cost of the delivered product will vary, depending
upon the source plant.
• Some orders require that all material must come from
the same plant to maintain consistency of materials or
appearance (for example, a pool deck). These orders are
called single-source orders.
• Some customers place two orders for the same day and
request that they be linked. This means that they want the
second order to begin only after the first order is completed.
• The customer will occasionally request a gap in the
delivery to account for a planned delay during the pouring
of the concrete.

3. The Environment

The environment in which concrete is delivered is very
dynamic. Obviously, the time it takes to travel from a
plant to a customer site (or vice versa) can vary signifi-
cantly within a day and from day to day. Traffic tie-ups
in the Northern Virginia area are likely and unpredictable.
Depending on the size of the variation, it can be difficult
to accurately coordinate fleet departure times for timely
arrival at customer sites. Additionally, customer behavior
can impact the efficiency of the delivery process. Some
common customer problems are:
• Customers may not be ready when the driver arrives
with the concrete.
• Customers are often unable to unload trucks in the
time that they specify.
• The customer seldom knows the exact amount of con-
crete that they will need (underestimating by as little as one
cubic yard could require an additional truck delivery).
• The customer can delay or cancel an order up until the
time that the first of the trucks associated with that order
is loaded.

These customer-created problems cause the driver and
truck to be in use for a longer period of time than is neces-
sary, thereby causing Virginia Concrete to incur an increase
in the cost of delivering concrete. Additionally, Virginia
Concrete encounters an opportunity cost resulting from the
inability of the company to use the truck on other deliver-
ies because it is delayed at the current customer. Together,
these costs have a significant impact on the profitability
of the company. Therefore, the ability to assess the cur-
rent state of all deliveries (to determine if the delivery is
on schedule or delayed) and to respond appropriately can
provide significant cost savings to the company.

Weather can impact the delivery process in a variety of
ways. The most obvious is that inclement weather impacts
the expected travel time and causes delays in deliveries.
Additionally, concrete cannot be delivered to many cus-
tomers when there is heavy rain or cold temperatures.

Another factor adding to the dynamic environment is the
breakdown of either trucks or plants. The breakdown of an
empty truck can result in delivery delays until the truck
is brought back into service or until the driver is placed
into a backup truck. The breakdown of a loaded truck has
additional costs associated with material replacement: late
delivery to the customer and removal of hardened prod-
uct from the drum of the truck. The breakdown of a plant
can create a significant disruption in the delivery plan for
the entire company. When this occurs, the company is pri-
marily interested in system recovery and places priority
on continuing all customer deliveries that are currently in
progress.
4. Similarities to Other Scheduling Problems

The process resulting from order taking, scheduling, routing, and dispatching by CSRs and dispatchers can be defined as the generation of a series of tasks that relate to each delivery. Looking at it in this way, a task relates to a single customer delivery and is defined by an origin, a destination, a starting time, and an expected duration. At first glance, this problem has many of the attributes of a vehicle-routing problem (VRP), where the capacity of the vehicle limits it from visiting more than one customer per trip. Factors such as multiple depots, delivery time windows, and a dynamic environment are complications to the classic VRP that have been investigated and reported by many researchers. There are, however, two intertwined aspects of the concrete delivery problem that set it apart from classic VRPs: the perishable nature of the product being delivered and the time-synchronized deliveries required by many customers.

In the 1950s, researchers observed that the temporal aspects of a problem could be modeled via a network flow model. This temporal model is now called a dynamic network. The dynamic network is a natural representation of the VRP, which seeks to find the minimum number of vehicles required to visit a set of nodes subject to capacity constraints. Dantzig and Fulkerson (1954) first used a dynamic network model when formulating a tanker-scheduling problem with dynamic networks. Shortly thereafter, Ford and Fulkerson (1956) discovered innovative methods for determining the maximum flow over a dynamic network. More recently, there have been many uses of dynamic networks to solve problems. Among them, Zawack and Thompson (1987) used dynamic networks to model dynamic traffic assignment, Oh and Haghani (1997) employed dynamic networks to generate a model for disaster relief management, and Bertsimas and Patterson (2000) utilized dynamic networks when rerouting aircraft.

As noted in Desrosiers et al. (1995), the dynamic model for the single-depot problem VRP can be extended into a multiple-depot problem in a straightforward manner. This is accomplished by adding a supersource and a supersink to the model.

The vehicle routing problem with time windows (VRPTW) seeks to find the minimum number of vehicles required to visit a set of nodes, subject to time windows and capacity constraints as described by Bard et al. (2002). The objective is to find the minimum number of tours, $K^*$, that does not exceed the vehicle capacity, $Q$, such that each node is serviced within its time window. Key elements of this problem include: (1) static customer demand, (2) known time windows, (3) each customer must be served by exactly one vehicle, and (4) the tours must start and end at the depot. Once the number of tours, $K^*$, is resolved, then at most $K^*$ vehicles will be needed. If any two tours are time disjoint, then one vehicle may satisfy both of the tours.

This formulation results in a complex mixed-integer problem (MIP) that is capable of generating the optimal solution to the VRPTW. The MIP is solvable in a reasonable amount of time for moderate-sized problems, but can be problematic if the problem is very large, as many dispatching problems tend to be. Consequently, as demonstrated by Bixby and Lee (1998) and Bard et al. (2002), much of the research in dispatching incorporates heuristic solvers. Both Bixby and Lee (1998) and Bard et al. (2002) presented arguments for implementing branch-and-cut algorithms to find an optimal solution, although neither addressed problems incorporating time windows. The research of Bixby and Lee (1998) and Bard et al. (2002) was motivated in part by Crowder et al. (1993), and Hoffman and Padberg (1991, 1993), who implemented branch-and-cut algorithms to solve large-scale integer problems.

Of particular interest in the Gendreau et al. (1999) paper was the fact that time window restrictions are handled as soft constraints. Early arrival at a customer site has the implicit penalty of vehicle idleness, so nothing was required. However, late arrival at a customer site results in the explicit penalty of increased cost. The amount of the penalty is dependent upon the tardiness of the vehicle. The penalty function should be balanced so that late arrivals occur if necessary, but they are not so commonplace that customer dissatisfaction is rampant.

More recently, a significant amount of research has focused on applying dynamic networks to aircraft, crew, and job-shop scheduling. Ahuja et al. (1993) formulate a scheduling problem on uniform machines by implementing a bipartite network with a source node, a sink node, a layer of nodes that represent each job, and a layer of nodes that represent a time window in which the job can be performed. Klabjan et al. (2002) apply a time-space network model to aircraft routing and crew scheduling, where the columns are generated dynamically. The size of the resulting MIP is daunting; hence, the problem is broken into separate crew-scheduling and routing problems. The problems are solved sequentially, with results of the crew-scheduling problem feeding into the routing problem.

The majority of the scheduling solutions currently in use focus on planning models. This paper addresses the solution of a real-time scheduling problem with perishable products in a time-dependent, dynamic environment. On a good day, 75% of the orders change and traffic is fairly predictable. On a bad day, 95% of the orders change, equipment breakdowns occur, and traffic disturbances are significantly beyond the expected range.

The problem that most closely resembles our dispatching problem is an airline’s problem of recovery after a major disruption caused by extreme weather conditions or other unusual occurrences (e.g., computer malfunction, labor walkout). As described in Clausen et al. (2005), in such circumstances, the airline industry cancels or delays many flights. After such an occurrence, the major goal is
to restore the schedule as quickly as possible to its prior state. The airline industry performs such massive system recoveries on an infrequent basis. On the other hand, in the concrete industry, the problem is that there are significant additions to given orders, new orders are arriving, and the existing orders are being changed—all in an unpredictable manner. For this application, we regenerate schedules every five minutes of every day due to the constant nature and magnitude of the changes to the system. Another unique dimension to this problem is the extreme perishability of the product. Most perishable items have a shelf life measured in days, whereas concrete has a shelf life of 2–3 hours. Additionally, if the concrete is not delivered to a customer and not disposed of properly, it can harden in the truck, resulting in a large capital expense. A final aspect of this problem is that the inability of customers to forecast their needs is far less predictable in the concrete industry than may be encountered in other industries.

5. Overview of the Solution Approach

To solve this problem, a decision-support tool was created that consists of both planning and execution modules. The tool is used to (1) assist CSRs and dispatchers in determining the feasibility of accepting additional orders, (2) determine the arrival times for drivers reporting to work, (3) generate the schedule for all orders, (4) assign drivers to delivery loads in real time, (5) determine the return plant for drivers after completing a delivery, and (6) schedule the load sequencing at each of the plants. This research describes the series of optimization models required to implement a decision-support tool, the implications of imperfect data, and implementation issues associated with real-time requirements.

Similar to Lourenço et al. (2001), this problem is decomposed into subproblems due to the complexity and size of the problem. Specifically, the problem is decomposed into planning and operational subproblems, then further decomposed based on processing mode (batch or real-time processing) to generate five components: the order-entry planner (OEP), the arrival-time planner (ATP), the next-day planner (NDP), the real-time planner (RTP), and the real-time dispatcher (RTD) (see Figure 2).

The order-entry planner (OEP) is phase one of the decision-support tool. The purpose of the OEP is to allow CSRs the ability to accurately estimate future workload so that they can determine the viability of new orders. To support this, the OEP generates a plan for future days, allowing for overbooking of both trucks and plants. When the CSR accepts a new order, the schedule is recalculated. The arrival-time planner (ATP) is phase two of the decision-support tool. The purpose of the ATP is to determine what time the drivers need to arrive for work the next morning. The ATP runs at the close of business for the day, and assumes that all orders for the following morning are known. The next-day planner (NDP) is phase three of the decision-support tool. The purpose of the NDP is to generate the best possible schedule for the next day. The NDP runs at the end of each day after all the orders for the next day have been taken and the ATP has determined the arrival time of the drivers. The real-time planner (RTP) is phase four of the decision-support tool. The purpose of the RTP is to advise CSRs regarding the feasibility of accepting an order for a later time that same day. The real-time dispatcher (RTD) is phase five of the decision-support tool. The RTD is required to continually update the schedule for orders currently in progress as well as for any orders that begin within the local time horizon under consideration (typically 1 ½ to 2 hours). The RTD does this on a continuous basis due to schedule changes that occur continuously but unpredictably throughout the day.

6. Representing the Dispatching/Planning Problem as a Time-Space Network

The initial deployment of the decision-support tool at Virginia Concrete consisted of the order-entry planner (OEP) and the arrival-time planner (ATP). At first, both of these components mimicked the behavior of the dispatchers and were truck based rather than demand based. The advantage of creating tools that mimicked tasks previously performed by the dispatchers is that the tool unburdened overworked employees. More importantly, it allowed us to prove that we understood the complexity of the business problem and the issues encountered by the schedulers. Having the confidence of the dispatchers proved essential when moving from truck-based to demand-based dispatching. Additionally, we were able to use the travel times that were continually updated via the GPS system rather than static travel times.

The truck-based OEP was an interactive tool that allowed CSRs to respond to overbooking by evaluating the impact of different customer delivery options on fleet usage. This tool immediately improved the scheduling skills of the dispatchers because they now had better information upon
which to make decisions. The OEP also introduced historical travel times to the dispatching team (they had been using static travel times previously).

The truck-based ATP was the first optimization-based tool created for Virginia Concrete, and it generated the arrival time of the drivers for the next day by imitating and automating the process of the dispatchers. A network flow model (as seen in Figure 3) was generated to determine the least-cost assignment of drivers to deliveries, from which the arrival times were then calculated.

The problem depicted in Figure 3 consists of assigning a supply of 40 trucks to the first customers of the day (who have a combined demand for 40 trucks). There are four layers of nodes: the supersource node, plant source nodes, customer nodes, and the supersink node. Arcs connecting the different layers represent possible movement of trucks over the network (some motion is notional, some motion is physical). Each arc is limited by a maximum flow capacity and has a nonnegative cost associated with movement over the arc. Arcs connecting the supersink and the plant source nodes represent the distribution of trucks to the plants where they parked at the end of the previous day (9 at Edsall, 19 at Dulles, and 12 at Fairfax). Arcs connecting the plant source nodes to the customer nodes represent the possible assignment of trucks from a plant to a customer. The flow over these arcs is limited to the number of trucks that will satisfy customer needs. The cost associated with these arcs is related to travel cost from the plant to the customer site. The last set of arcs connects the customer nodes to the supersink node. The maximum flow on these arcs is set to the number of trucks that satisfy customer requirements. Flow-balance constraints (requiring the sum of the flow into a node equal the sum of the flow out of the node) on the customer nodes ensure the proper number of trucks are assigned to each customer.

After solving this min-cost network flow model, we have determined the number of trucks from each plant that will be assigned to each customer. With this information, the process of determining the arrival times of the drivers involves the simple process of backtracking from the scheduled delivery time, accounting for staging time, travel time, wash time, and load time. For example, if Customer 132 requested a 7:00 a.m. delivery with a 10-minute staging time, then the truck must arrive at the site at 6:50 a.m. If the model determined that the truck should come from the Dulles plant (with 23 minutes travel time), then the truck must depart the plant at 6:27 a.m. to arrive on time. If the Dulles plant has an average wash time of 10 minutes and an average load time of three minutes, then the driver must be sitting in the truck ready to load at 6:14 a.m. If the model assigned two trucks from the Dulles plant to Customer 132, and Customer 132 requested a 15-minute interarrival rate on the trucks, then the second driver must be in a truck ready to load at 6:29 a.m. for a 7:15 a.m. on-time delivery of the second load.

With the experience gained from developing this model in hand, we began developing the demand-based components of this support tool: the next-day planner (NDP),..
the real-time planner (RTP), and the real-time dispatcher (RTD). These three modules address the scheduling and dispatching of orders for the current day. The successful utilization of a network flow model in the ATP allowed us to consider the use of network-based models for the demand components.

A time-space network representing the delivery of concrete is displayed in Figure 4. In this example, one load of concrete is to be delivered to a customer at 9:00 a.m., represented by the pour arc in the center of the figure. Two alternative plants are included in the diagram of the network to demonstrate some of the options available to the dispatcher. The vertical axis represents location, whereas the horizontal axis represents time. Thus, the top line of nodes in the figure represents a specific plant at discrete time intervals (in this example, five minutes), whereas the nodes along the bottom row represent an alternative plant (with six-minute time intervals). The choice of the time interval for a plant is equal to the time required to load concrete on a truck at the plant (plant equipment can vary—as does the load time). The arcs connecting a plant to itself at the next time step allow a vehicle to remain at the plant in an idle mode. There is unlimited capacity on these plant idle arcs.

In addition to the plant nodes (that allow trucks to remain at a plant over time), there are also plant load nodes. Flow traversing the arc connecting a plant node to a plant load node represents the act of loading a truck with concrete for a particular customer, and the time associated with that action. This arc is called the load arc. The arc leaving a plant load node represents the time associated with washing the truck at the plant. The next arc along this path represents transit time to the customer site. After arriving at the customer site, the truck must get in the proper position to unload the concrete—a process called staging. The next arc in the path is the pour arc. Flow over this arc represents pouring concrete at the customer site. After pouring concrete, the truck must again be washed, and the time required for this action is accounted for when flow is sent over the wash arc. Finally, there are arcs representing possible return trips for the truck. All arcs except the pour arc have a cost (negative profit) associated with the time required for traveling, waiting, or performing such operations as washing or loading. The only positive profit arcs are the pour arcs that represent the revenue of delivering the order to the customer.

The capacity on all arcs in this representation (other than the plant idle arcs) is equal to one. This results from the fact that a plant can only load one truck at a time, and from the fact that each pour arc represents delivery of one truckload of concrete. For this reason, the wash arc, transit arc, and staging arc can be compressed into one arc that accounts for all of these actions. This new arc is called the to transit arc and can be seen in Figure 5. Likewise, the wash arcs at the site can be compressed into the return transit arc. This new arc is called the from transit arc and can also be seen in Figure 5. Additionally, the figure contains arc capacity and arc revenue/cost.

Although this pure time-space network formulation could easily be expanded to handle 10–12 plants and 500–600 loads of concrete per day, there was one aspect of the problem not yet accounted for by the formulation—Virginia Concrete cannot deliver all orders as requested. As is very common in the concrete industry, Virginia Concrete practices overbooking (accepting more orders than they can...
handle based upon their resources). As a consequence, generating a feasible solution to the scheduling problem as described above may not be possible. To avoid this problem, multiple delivery alternatives are generated for each order that include slipping the start time of the order and stretching the interarrival rate of the trucks. The time-space network shown in Figure 6 represents five delivery alternatives for one customer who has requested four truckloads.

**Figure 6.** Time-space network.

*Includes time for plant washdown, transit time, and staging.

**Includes time for site washdown and transit time.
of concrete. In this representation, we have two plants, each of which can serve as the source and/or destination of the trucks. This very small problem results in a network with 220 nodes and 328 arcs. On a typical day, Virginia Concrete will deliver 500–600 loads to approximately 100 customers from 10–20 plants. Eleven alternatives are created per customer, and three to five plant alternatives are included for the source and/or destination of each load. Additionally, deadhead arcs (arcs from plant to plant that represent an empty truck moving from one plant to another) are included to allow for relocation of trucks. Finally, the network requires source and sink nodes that represent the start and end conditions for the day. When considering the entire day, a typical day is modeled as a network having approximately 25,000 nodes and 175,000 arcs.

Having implemented the time-space network for the NDP, we realized that the same methodology could be used for all five of the models: the NDP, the RTP, the RTD, the OEP, and the ATP. Thus, once the time-space network model was built, we redesigned the OEP and ATP modules, which were truck-based models, so that they too became demand-based models. We also realized that we could reuse much of the network generation when moving from one model to another and when we needed to re-solve the same model with changes in the data. We next describe the mathematical formulation and then describe the differences among the models.

7. Mathematical Formulation

We present the mathematical formulation of the time-space network for the NDP.

### Input Data

- \( n \) number of nodes.
- \( n_t \) number of trucks available during the time period.
- \( n_p \) number of plants.
- \( p_z \) number of trucks that must start and end the day at each plant (\( z = 1, n_p \)).
- \( c_{ij} \) capacity of the arc connecting node \( i \) to node \( j \).
- \( v_{ij} \) value (revenue or cost) obtained by sending one unit of flow over the arc connecting node \( i \) to node \( j \).
- \( s \) denotes the source node.
- \( t \) denotes the sink (terminal) node.

### Defined Sets

- \( A \) set of all arcs in the time-space network (arcs will be defined by the two nodes that the arc connects).
- \( N \) set of all nodes in the time-space network.
- \( S \) set of arcs flowing out of the source node (\( S \subset A \)).
- \( T \) set of arcs flowing into the sink (terminal) node (\( T \subset A \)).
- \( P_{sp} \) set of arcs that flow from the source node into plant \( p \) (\( P_{sp} \subset S \)).
- \( P_{pl} \) set of arcs that flow from plant \( p \) into the sink (terminal) node (\( P_{pl} \subset T \)).
- \( I_n \) set of arcs flowing into node \( n \).
- \( O_n \) set of arcs flowing out of node \( n \).
- \( A_b \) set of alternatives for order \( b \).
- \( P_{ab} \) set of “pour arcs” associated with alternative \( a \) and order \( b \).
- \( SS_{abp} \) set of “to-transit arcs” associated with traveling from plant \( p \) to the customer site associated with order \( b \) using alternative \( a \) and pouring the \( q \)th pour of that order.
- \( C_b \) set of plants that are available to supply concrete to order \( b \).

### Decision Variables

- \( X_{ij} \) flow over the arc connecting node \( i \) to node \( j \).
- \( Y_{ab} = 1 \) if alternative \( a \) is selected for order \( b \), \( = 0 \) otherwise.
- \( W_{bp} = 1 \) if plant \( p \) is selected as the source for order \( b \), \( = 0 \) otherwise.

#### The Real-Time Dispatching Problem Formulation:

\[
\text{Maximize } \sum_{i=1}^{n} \sum_{j=1}^{n} X_{ij} v_{ij} \tag{1}
\]

subject to

\[
\sum_{(i,j) \in S} X_{ij} = n_t, \tag{2}
\]

\[
\sum_{(i,j) \in T} X_{ij} = n_t, \tag{3}
\]

\[
\sum_{(i,j) \in P_{sp}} X_{ij} = p_z \forall \text{ plants } p, \tag{4}
\]

\[
\sum_{(i,j) \in P_{pl}} X_{ij} = p_z \forall \text{ plants } p, \tag{5}
\]

\[
\sum_{(i,j) \in P_{sp}} X_{ij} - \sum_{(j,k) \in O_n} X_{jk} = 0 \forall \text{ nodes } n \text{ such that } n \neq s, t, \tag{6}
\]

\[
\sum_{a \in A \cap A_h} Y_{ab} \leq 1 \forall \text{ orders } a, \tag{7}
\]

\[
X_{ij} \in P_{ab} = Y_{ab} \forall \text{ alternatives } a \text{ and orders } b, \tag{8}
\]

\[
X_{ij} \in SS_{abp} = W_{bp} \forall \text{ pours } q \text{ in alternative } a \text{ in single-source order } b \text{ for each plant } p, \tag{9}
\]

\[
\sum_{p \in C_b} W_{bp} \leq 1 \forall \text{ single-source orders } b \text{ for each plant } p, \tag{10}
\]

where

\[
Y_{ab} \text{ binary, } \tag{11}
\]

\[
W_{bp} \text{ binary, } \tag{12}
\]

\[
0 \leq X_{ij} \leq c_{ij}, \tag{13}
\]

(1) The problem seeks to maximize profit. All arcs other than the pour arcs have a cost (negative profit) associated with the time required for traveling, waiting, or performing
such operations as washing or loading. The pour arcs have positive profit and capture the value of the delivery.

(2) The number of trucks flowing out of the source equals the number of trucks available company-wide.

(3) The number of trucks flowing into the sink equals the number of trucks available company-wide.

(4) The number of trucks flowing out of the source to each plant source node equals the number of trucks available at the plant.

(5) The number of trucks flowing into each plant sink node equals the number of trucks available at the plant.

(6) The flow into any node (exclusive of the source node and sink node) equals the flow out of the node.

(7) The sum of customer alternative variables is less than or equal to one. Note that the inequality signifies that there is not a requirement to deliver concrete to all customers. There could be circumstances (such as a drastically overbooked day with no cancellations, or the breakdown of a vital plant) that impact the ability to deliver all of the requested concrete. In these cases, it is undesirable to encounter infeasibilities that impact the ability to deliver a solution to the dispatchers. Instead, it is preferable to generate the best answer possible and inform the dispatchers (via the user interface) about the problem in the hopes that they will be able to obtain some resolution.

(8) Pour arcs associated with a specific alternative can only be positive when the associated 0-1 alternative variable is equal to one. Conversely, if the alternative is chosen, then the flow over each pour arc associated with this alternative must be equal to one. This constraint enforces the concept that if delivery to a customer has begun, then all of the concrete that has been requested must be delivered—and it must be delivered in a time-synchronized fashion.

(9) For single-source orders, the same plant must be the source of all the deliveries made to the customer.

(10) For single-source orders, only one plant may be selected as the source.

(11) The alternative variables are binary.

(12) The single-source variables are binary.

(13) The flow over each arc is nonnegative and limited by the capacity of arc.

Note that constraints (2) and (3) are implied by (4) and (5) and can therefore be removed from the formulation. We have included them in the formulation for clarity of exposition.

8. Reformulations, Results, and Run-Time Issues

All of the results listed here were generated with CPLEX 8.1 on an HP server with dual processor 2.80 GHz CPUs running Microsoft XP server. To speed the solution of the root node, the Barrier method was selected.

8.1. Reformulation

The time-space network formulation with multiple delivery options is the foundation for all of the required models. The difference between the models is the time period under consideration, the time allocated to run the model, and the known information. We now describe each module, highlighting the differences between the formulation defined earlier and the formulation required to successfully meet all requirements.

The order entry planner (OEP) provides a schedule for the next two to three days to the CSRs, thereby allowing them to make more informed decisions when customers request service. To ensure that the CSR has the best information possible, the OEP would need to regenerate the schedule for a particular day as soon as any new order for that day is requested, or an existing order for the day is changed. However, the computational time required to regenerate a schedule from “scratch” makes such calculations unreasonable given that the schedule is likely to change many times prior to the delivery day. To balance the needs of the CSR with computational time requirements, the OEP is scheduled to run every hour, on the hour, from 9:00 a.m. to 5 P.M. At each iteration, the OEP regenerates the schedule for the next two days, evaluating four to six alternatives for each customer delivery. The OEP typically takes 5–20 minutes to regenerate a schedule for one day. (Note that driver arrival times for future days have not been calculated when the OEP runs.) These calculations are done to simplify the problem; the OEP is formulated so that all available drivers are available one hour before the earliest delivery for the day. One reason that the OEP can solve to 1% optimality in this time period is that we feed it the solution from the prior run of the same day. Often, there are only one or two changes from the previous hour and much of the scheduling remains the same.

The arrival-time planner (ATP) determines the arrival time of the drivers for the next day and is run at the end of the day. Because this information needs to be disseminated to the drivers, the ATP needs to generate an answer relatively quickly (10–20 minutes). This time requirement is rather restrictive, especially when considering that drivers can arrive as early as 3:00 A.M. or as late as 9:00 A.M. Consequently, the network model must contain arcs that connect the driver source node to plant nodes at each possible load time between the earliest and latest arrival time, thereby allowing the model to determine the time that the drivers will arrive for work in the morning. A model that includes 11 possible delivery alternatives and all possible arrival times results in a problem with far more potential arcs than is necessary. Instead, we limit the number of delivery alternatives to no more than six, and restrict arrival times to vary up to 50 minutes prior to a planned load time for that delivery. Additionally, we only incorporate orders that begin before 10:00 A.M. (Orders occurring later in the day will have little impact on what time the driver needs to arrive for work in the morning.)
The next-day planner (NDP) generates a baseline schedule for the next day and runs overnight. The NDP utilizes the arrival times determined by the ATP along with the order data at the close of that business day. The NDP evaluates 11 alternatives for each delivery, and typically takes between 30 minutes and four hours to run, depending upon the amount of concrete to be delivered the next day. This model is required to generate a solution within 1% of optimality.

The real-time dispatcher (RTD) runs every five minutes and regenerates the schedule for the next one to two hours (the local time horizon defined by the user) based upon the current status of the fleet and the current status of each order. The time-space network must be modified so that any order that is in progress must be satisfied (the inequality in constraint set (1) becomes an equality constraint for any in-progress orders). The network formulation includes 11 delivery alternatives for each order not yet begun, and six delivery alternatives for any order that is currently in progress.

The real-time planner (RTP) regenerates the schedule for the remainder of the day so that CSRs can make more informed decisions when customers request service for the current day. As is the case of the OEP, one might want to regenerate the RTP schedule as soon as a new order for today is accepted, or an existing order for today is changed. As with the OEP, to balance the needs of the CSR with computational time requirements, the RTP is scheduled to run once per hour, taking the current status of the fleet and of all in-progress orders into consideration when regenerating the schedule. The network model representation includes four to six delivery alternatives for each order, and excludes all deliveries scheduled for the next two hours (these are being regenerated by the RTD every five minutes). Coupling the schedules generated by the RTD along with the schedule generated by the RTP gives the CSRs sufficient information to understand the current status of the orders and fleet to determine if orders should be taken.

8.2. Results

The NDP created the largest models, averaging approximately 175,000 columns and 25,000 rows, with 1,500 to 2,500 integer variables. These models took from 10 minutes to four hours to solve to within 1% of optimality, with the run time dependent upon the maximum amount of concrete requested for delivery (see Figure 7). As seen in the figure, the run times increase as the yardage increases, with some notable exceptions. The model with the smallest amount of concrete to be scheduled was one of these outliers, halting when the four-hour time limit was encountered (although an integer solution with a 1.06% integrality gap was successfully generated). The run time was highly dependent upon the number of nodes traversed in the branching tree before finding an integer solution within the bounds specified (as is seen in Figure 8).

Although we had hoped that we could prove optimality in the overnight runs of the NDP, optimality could not be achieved given a maximum time limit of eight hours. To date, we have run many of the NDP models related to days with more than 3,000 cubic yards of concrete for a minimum of 24 hours and have not been able to prove optimality. We have been able to obtain solutions within 1% of optimality every day. The 1% optimality has always been achieved within the four-hour time limit required by the company.

The RTP and OEP are very similar in both scope and formulation; hence, results from the two models will be reported together. The RTP schedules the remainder of the current day and creates models that average 90,000 columns and 12,000 rows; the OEP schedules a future day and creates models that average 120,000 columns and 18,000 rows. The RTP typically solves in less than five minutes, whereas the OEP typically solves in 5-20 minutes to within 1% of optimality. Remember that the RTP considers only a portion of the day and has relatively good information from prior optimization runs. On the other hand, the OEP is considering an entire day and may have inserted more than one large order that significantly alters the current solution.

Figure 9 plots the quantity of concrete scheduled for delivery versus run time. Two different run times are included in the plot: the run time to solve the root node relaxation (hollow triangles) and the run time for generating an integer solution (filled squares) within the integrality
gap tolerance of 1%. The run time required for the generation of both the root node solution and the integer solution increases as the quantity of scheduled concrete increases. The root node solution has an almost linear relationship with the run time for the quantities scheduled, whereas the integer solution time has an exponential relationship with the run time for the quantities scheduled. In fact, the “knee” in the curve seems to occur at approximately 6,000 cubic yards of concrete scheduled.

The RTD schedules the next, 1 1/2 to 2 hours of the day and considers many alternative ways of handling each order. The RTD model has approximately 80,000 columns and 10,000 rows during the busiest time of the day, and 20,000 columns and 2,000 rows during slower times of the day. The size of the model is a function of the quantity of deliveries that fall within the time window under consideration. As is seen in Figure 10, the morning is generally the peak time of day for concrete deliveries, with another peak occurring slightly after 12:00 p.m. Because the local time horizon was set to two hours in each of these runs, the post-lunch orders impact mostly the run times for model runs between 11:00 a.m. and 1:00 p.m. However, the run times decrease as the crunch time approaches because we have good starting solutions from the previous runs.

Figure 11 displays the run time required by the model to solve both the root node—including heuristics and cutting planes—and the integer problem as a function of time of day (the solution time for the root node is the lower of the two lines in the figure). There is little separation between the run times, indicative of the fact that roughly 90% of the time CPLEX reached the optimality tolerance (1%) at the root node. Even when branching is necessary, the number of nodes evaluated was typically under five nodes to reach a 1% tolerance level.

Finally, Figure 12 displays the run time of the RTD as a function of the quantity of concrete scheduled for delivery. Because the RTD focuses only on deliveries that must occur during the next two hours, the quantity of concrete scheduled by the RTD is typically 2,000 cubic yards or less. This amount generates a moderate-sized model that can be solved in less than 60 seconds. As seen in Figure 12, there is a linear relationship between the run time and quantity considered.

8.3. Run-Time Issues

The most difficult problem encountered in this modeling effort was dealing with real-time aspects of the problem and most significantly, how to handle the unanticipated changes that impact all orders thereafter. For example, when an in-progress order is behind schedule due to a shortage of trucks, the model may recommend diverting trucks that had previously been designated for an order about to begin. Although this may satisfy the needs of the in-progress order, an order that is about to begin must now be slipped or stretched due to the shortage of trucks. We are, in effect, transferring the problem from the current order to a later order.
A second problem is that the data may change significantly while the solution is being calculated. For example, in the time interval between the data snapshot and schedule completion, a truck may be placed in “shop” status (meaning that it needs some mechanical adjustments and is unavailable for an unknown amount of time). The schedule likely made use of this truck, and until the problem is re-solved, the schedule will be inaccurate. As a result, the solution generated is outdated before it can be used. Re-solving the model every five minutes helped the dispatchers deal with these inconsistencies.

A third problem that had to be addressed was the fact that often the solution generated contained assignments that were outside of what was considered normal by the dispatchers. From a dispatching point of view, many of the deliveries have specific alternatives that are preferable to others. However, removing alternatives that are less desirable could generate an infeasible problem, which is unacceptable. Therefore, many alternatives are generated for each delivery in the network flow model, with bonuses included to induce desired behavior and penalties included to discourage undesired behavior. By judiciously setting the penalties, we were able to force the model to choose more appropriately among the feasible alternatives, and by so doing, gained the confidence of the dispatchers. Furthermore, the introduction of penalties and bonuses had the added benefit of decreasing the degeneracy in the model and thereby aiding the solver in generating a solution faster.

8.4. Implementation Issues

We present a number of implementation issues encountered in real-time problems as contrasted with planning problems. Each of these issues was resolved by initiating discussions with the dispatching team and management and determining precisely how best to modify the model’s behavior. Many interface changes were made to the model so that the dispatchers could supply the model with more information upon which to base decisions. We now present a discussion of only the more significant changes that occurred.

One problem encountered was that of feasibility. When executing either the NDP or the OEP, one may discover that there are no feasible solutions to the problem. This infeasibility results from the fact that more orders have been placed than can conceivably be fulfilled by the available trucks and drivers. To overcome this seemingly impossible situation, we add simulated trucks to the model (called phantom trucks). Through experimentation with parameters, we determined penalties to apply to these phantom trucks to assure that they were only used when absolutely necessary. Adding a cost that scaled upwards as the number of phantom trucks increased allowed us to provide important warnings to Virginia Concrete as to when to borrow trucks from sister companies, when to carefully monitor order placement, and when to alert their customers to significant time delays.

Similar experimentation was necessary to determine whether slipping orders or stretching orders was a better policy for the company. We examined prior histories and had detailed meetings with management to determine an overall strategy. We created a strategy that would have the orders arrive on time as often as possible, with the stretching of an order being a better alternative than the slipping of an order. This strategy was chosen because dispatchers believe that beginning a delivery 30 minutes late is much more noticeable to a customer than increasing the interarrival rate of the trucks from 10 minutes to 15 minutes. An interface that allows dispatchers to modify the rules was also developed. This allows the dispatcher to describe a preference while assuring that feasible solutions are generated.

A second modification relates to highly valued customers. Virginia Concrete highly values certain customers due to the high volume of business they provide. A new customer is also considered a highly valued customer. Virginia Concrete prefers that the decision-support tool generate a schedule as close as possible to what these highly valued customers have requested. Therefore, when a slip (delaying the start time of the delivery) or a stretch (increasing the interarrival rate of the trucks) was required to generate a feasible solution, Virginia Concrete would prefer that other customers incur most of the delay. This issue was resolved by creating an interface for the dispatchers to inform the decision-support tool which customers were preferred over others. A penalty is then added to any alternatives that include a slip or a stretch for the preferred customers.

Another modification was made when we realized that trucks were often arriving early to the customer site. We had thought that an early departure from the plant would be preferred (all other things being equal) because the schedule is generated using average travel times, and early departures would help compensate for the inevitable variations in the averages. We therefore thought it best to allow more travel time when possible and have trucks sit idle at the customer site rather than at the plant. However, dispatchers requested that we put a limitation on such early arrival of trucks to be no more than 10 minutes. This limitation was easily imposed on the plant load arcs created when generating the model.

A fourth real-time issue encountered was dealing with rules associated with in-progress orders that, if enforced without any slack, would not allow a feasible schedule. This arose when two supposedly inviolable constraints were encountered simultaneously. For example: (1) once delivery of concrete has begun to a customer, the job cannot be stopped until it is complete; and (2) a single-source order must have all concrete come from the same plant. In this instance, delivery of concrete began to a single-source order from a particular plant. Before the order was complete, the plant broke down. Completion of the single-source order was not possible because the plant was no longer available.
Virginia Concrete determined that completing the delivery should override the initiating plant constraint. Thus, determining the precedence order of conflicting constraints was necessary. With this precedence information, a soft constraint was instituted that ensures that a plant would be the source of all deliveries to a single-source order if the plant remained open. If the plant closed down, an alert would be sent to the dispatchers and a different plant would be assigned to continue the delivery.

A similar concern relates to the interarrival rates of the vehicles at a customer site. Virginia Concrete’s customers are often optimistic regarding their ability to unload vehicles, which directly relates to the interarrival rate of the vehicles. However, it was also true that some customers require several trucks of concrete to get into a rhythm, and responding too quickly to their early behavior can be an overreaction. During testing, we identified customers that significantly overstated their capabilities (for example, a customer who requested 10-minute spacing was performing at an average time of 38 minutes). Altering the interarrival rate of vehicles for these customers was vital to successfully meeting demands placed upon the company by other customers. However, if the customer performance improves as the order progresses, we want to be able to readjust the schedule accordingly. Consequently, for each customer, a rolling average of the last three truckloads is now calculated to determine the interarrival rate for future scheduling of deliveries.

A similar approach is taken when dealing with bonus pours. Because many customers cannot estimate exactly how much concrete they will need, two values are recorded when the order is taken: the amount of concrete ordered, and the maximum amount the customer thinks they may need. The trucks required to deliver the difference between these two values are referred to as bonus pours. The fact that bonus pours become actual deliveries as often as not requires us to account for the delivery of bonus pours when generating a schedule. However, Virginia Concrete will only ship concrete to a customer up to the amount ordered by the customer. Often, it is not until the last scheduled truck is finished that Virginia Concrete learns that the customer will need more concrete. Therefore, a gap (30–60 minutes) is inserted into the schedule between the delivery of the ordered amount and the delivery of the bonus pours. In this way, we attempt to account for the fact that the additional deliveries may or may not occur. When the ordered amount is increased, the additional yardage is scheduled for delivery and is given a high priority so that it will be delivered as soon as possible. Until the ordered amount is increased, the bonus pours continue to be slipped into a later time. These pours are removed only when the order is closed.

9. Conclusions

Concrete is a perishable commodity; on a hot summer day, concrete can harden in the truck within two hours, not only making the product unusable, but destroying the drum of the truck. In addition, because of the nature of the construction industry and the inexact science of predicting required quantities, over 90% of the orders change during the day of delivery. Adding to this variability is unpredictable weather and sudden massive traffic tie-ups in urban areas. Finally, providing directions to some customer sites is complicated by the fact that the actual existence of roads can be variable in and around a construction site and the exact location of the delivery is difficult to specify without addresses.

This paper presents a methodology for the ordering, scheduling, and dispatching of concrete under these extreme conditions. Besides the optimization methodology employed, an easy-to-use decision-support tool was designed that allowed order takers, plant managers, schedulers, and dispatchers to better manage the business of delivering concrete and aggregates to customers. We began this project believing that only heuristic approaches would be feasible for the real-time components of this project because of the number of times the problem would need to be re-solved and the short timeframe for reacting to changes in the schedule. We found, however, that with careful problem formulation and a sophisticated optimization tool (CPLEX), we could use exact combinatorial optimization techniques and provide answers to the dispatchers that were within 1% of optimality virtually all of the time.

The problem was divided into its associated natural pieces: the taking of orders (including the feasibility of accepting additional orders for same-day delivery), the determination of arrival times for drivers reporting to work, the scheduling of orders, the real-time assignment of drivers to delivery orders, the dispatching of these drivers to customers and back to plants, and the scheduling of truckloading activities at the plant. The decision-support tool for each component was developed and then tested by sitting side by side with the dispatchers, schedulers, order takers, and plant managers to assure that the system provided them with the information that they required.

Although not our original intention, the process of creating an optimization-based support tool resulted in the reengineering of the dispatching process. In the ready-mix concrete industry, much of the concrete that is delivered requires a time-synchronized deployment of several trucks to a single customer site. The standard dispatching method is referred to as truck-based dispatching, where each job is assigned a specified number of trucks and a specific plant as the source of the product. These trucks are not reassigned until all of the concrete for the job has been delivered. By demonstrating that this static allocation of trucks and plants to a given job is inefficient, the company was willing to test and then accept the method recommended in this research, that of demand dispatching, whereby trucks are assigned a specific delivery (as opposed to an entire job) when they enter a yard of a plant. When a driver completes a single delivery, he is told which plant to return to, and when he
arrives at the plant, he is then assigned the next load. Thus, trucks move from plant to customer to plant, often moving from one customer to another and from one plant to another in the process. Dispatching trucks in this intelligent, responsive fashion results in a complex, intertwined movement of trucks throughout the day, which we have labeled “the dance of the thirty-ton trucks.” This behavior yields better solutions than truck-based dispatching (which is simply a special case of our general formulation).

Clearly, with this more complicated approach to scheduling, the dispatchers needed a decision-support tool that not only prescribes schedules for the day, but also supplies graphical presentations of these schedules so that they could understand why this reengineering would better serve the company.

The most important component of this research is the real-time dispatching tool that uses a time-space network to re-solve a very large combinatorial optimization problem every five minutes. This time-space network allowed deliveries to be stretched (the interarrival times of deliveries was increased) and/or slipped (the start time of a job was delayed). We encouraged the company to store information that indicated the quality of each customer. In this way, the model was capable of assuring on-time deliveries to their best customers. The model could also delay future (same-day) truck deliveries to sites where the customer was not ready for the first phases of the delivery. The model is capable of reacting quickly to plant closures, truck breakdowns, and traffic issues.

The success of this project was dependent upon the accuracy of truck positioning and status data. Each truck has a GPS system, a sensor, a CPU, and a wireless communication device. This equipment reports exact coordinates and status information to the dispatching center. In addition, Virginia Concrete created a new information system that tracks all aspects of the business, such as the rate at which various plants load concrete onto the trucks, the status of the inventory at each plant, and the variation in time to and from plants and customers based on time of day. The optimization suite is fed this information, reoptimizes the schedule, and reloads the changed schedule in less than five minutes.

Not surprisingly, the complexity of this project revolved around the fact that we were concerned with a real-time scheduling tool rather than a planning tool. Planning tools do not concern themselves with the minutiae of the minute-to-minute changes in operations. Real-time scheduling, on the other hand, cannot ignore the fact that a given truck is out of service because of an unscheduled break by a driver, a flat tire, or traffic on a main thoroughfare. Without such attention to emergent conditions, the project would not have been successful. Conditions such as the unexpected arrival or removal of drivers, trucks, and even plants from the system are handled automatically. The system must be able to overcome seeming infeasibilities that result when, for example, a CSR accepts a large order into the system on an already overloaded day. In these cases, the company borrows trucks from other companies, slips and/or stretches orders, or, as a last resort, reschedules orders by calling customers and negotiating changes. In this application, recognizing and reacting to these changes is critical.

This paper describes the main aspects of the scheduling and dispatching of concrete. Much of the detail of designing a user-friendly decision-support system and the necessary training and refining of such a tool are discussed only briefly. However, when one looks at the overall time expenditure on this project, we found that it was the human interface issues that required the greatest amount of time to resolve. However, without this attention to the needs of the dispatchers and CSRs, we doubt that the project would have succeeded.

Now, because of the success of this project, the decision-support system will be used not only by Virginia Concrete, but will also be installed throughout all of the companies owned by Florida Rock (the parent company of Virginia Concrete), thereby expanding the number of trucks dispatched from 125 to 1,400, and the number of plants from 10 to 150. Because the corporation has multiple companies located in various regions, we foresee this tool being used for scheduling of up to 250 trucks and 20 plants at any of their given dispatching centers. During the review process for this paper, two of the companies (Virginia Concrete and Cardinal Concrete) merged and are now being treated as a single entity. This merger has resulted in a company with 250 trucks and 20 plants, delivering up to 14,000 cubic yards of concrete per day.

As seen in Figures 7 and 9, there existed a “knee” in the run-time curve at approximately 7,000 cubic yards of concrete. As expected, none of the four planning models were able to generate a solution within the allocated time even when the optimality requirement was reduced from 99% to 95%. In fact, many times they were unable to generate any integer solution within the required run time. The OEP run time increased from a maximum of 30 minutes to a maximum of 6–8 hours. The NDP run time increased from a maximum of 4 hours to over 24 hours. The ATP run time increased from a maximum of 15 minutes to 2 hours, and the RTP run time increased from a maximum of 15 minutes to a maximum of 90 minutes. Only the RTD (which was generating a solution for the next 1 1/2 to 2 hours of deliveries totaling no more than 4,000 cubic yards) was able to stay within its time budget of 1–2 minutes.

To overcome these run-time issues, we first investigated running with faster hardware (3.06 GHz) and with the latest release of CPLEX (9.1). Although we were able to decrease the run time by 10%–30%, this was insufficient to meet our requirements. Hence, design changes were necessary to overcome the increased problem size (up to 40,000 rows and 160,000 columns) resulting from delivering up to 1,500 loads per day with up to 250 trucks.
The first consideration in our design change was to take advantage of the fact that the solutions from one run of the planning models to the next were closely related. As a result, we wanted to use the most recently generated schedule to “feed” the current run. The new “IP warm start” feature in CPLEX 9 gave us a mechanism to implement this design change. As a result, each OEP run utilizes the most recent OEP-generated schedule as a warm start. Likewise, the ATP and NDP use the most recently generated schedule from the OEP. The first run of the RTP uses the NDP-generated schedule as a warm start, and each successive RTP run uses the most recent RTP-generated schedule as a warm start.

However, the CPLEX warm start feature requires a feasible solution to the current problem as input. Due to cancellations and changes to previously accepted orders, our most recent solution is often not feasible for the current model. We therefore created an exchange heuristic that starts with the prior run’s solution, and works to adjust this into a feasible solution for the current run. Once feasibility is obtained, a tabu-search heuristic is employed to improve the solution. We stop this procedure when we see no improvement over many iterations or when three minutes have elapsed. The addition of the warm start along with the heuristic has an additional benefit: We are now guaranteed to generate an integer solution (although perhaps not within the optimality bounds specified earlier) to the problem within the allotted time.

With these design changes implemented, the planning models are capable of generating schedules within the allotted time for the merged company, although the optimality goal is not always obtained. The OEP continues to run each hour, and typically generates a solution to within 1% of optimality in no more than 40 minutes. Making similar changes to the ATP brought the maximum times down (even though the problem size increased significantly) from the previous 20 minutes (for the smaller problem) to 10 minutes for the combined-company model. We obtained a similar result for the NDP that runs in no more than two hours rather than the prior four hours. We provide figures comparable to those presented earlier for the larger merged problem in the appendix.

Once there is significant geographic separation, trucks and plants cannot be shared and the problems become disjoint problems. Because Virginia Concrete and Cardinal Concrete were two of Florida Rock’s largest companies before their merger, we are confident that our current tool is capable of being used throughout the Florida Rock system.

A final proof of success of this project is that Virginia Concrete is sufficiently convinced of the importance of this research and its application that they have begun to promote it as a “best practice” throughout the concrete industry. They highlighted this project at both the World of Concrete (2003) and ConAgg (2002) industry conventions in Las Vegas, and they are in the process of implementing this tool throughout the corporation.
Figure A.5. RTD run time vs. time of day.

Figure A.6. RTD run time vs. quantity scheduled.

Endnotes

1. Later runs, using CPLEX 9.1 on larger problems that occurred after the merging of Virginia Concrete and Cardinal Concrete, can be found in the appendix.

2. The requirement that all concrete come from the same plant occurs on architectural jobs (e.g., swimming pools, patio floors) where one must maintain the same dye lot to assure consistency of color or texture.

References


Dantzig, G., D. R. Fulkerson. 1954. Minimizing the number of tankers to meet a fixed schedule. Naval Res. Logist. Quart. 1 217–222.


