

Optimal Loading Scheduling

Miguel Martínez, Jorge González, Jesús Baquedano, Ramón Galián
Decision Support Systems,
Peña Santa 21, 28043 Madrid, Spain

The main purpose of this project is to find the optimal scheduling for all loading activities that have to be carried out in the central warehouse in order to cover all daily delivery necessities. The central warehouse is a compound of three different depots, all of them with different load capacities, working shifts and stocks. In addition, there are shuttles that can move merchandise between storehouses. The project uses Ilog Schedule and Ilog Solver to find a solution that makes a load appointment for each load (assigning location and time) matching all customer restrictions.

The Optimal Loading Scheduling project (OLS) has been developed by DSS for a large consume products company. The purpose is to assign loading bay and loading time for every truck that delivers nationwide from the central warehouse. This way, the company can avoid the extra costs associated with the waiting time of trucks delayed in a queue.

The loads are groups of several orders either for the same or different customers, and they must be delivered on time. This property is known as service quality. The major project objective is to maximize the service quality, in other words, to maximize the number of orders that are delivered on time. Each order has associated with it a loading time window, calculated so as to ensure that the order will arrive at the customer on a compromised date. When several orders are combined to be loaded in the same truck, the loading window is recalculated to meet all commitments.

Another relevant property of any load is the composition: the different products and the exact quantities of them that have to be loaded. A restriction imposed is that any load has to be done in a single bay. Sometimes there is not enough stock to complete some load in a single depot, and it becomes necessary to transfer stock from one depot to another. The number of shuttles needed to complete the transfers is also limited, and their use involve an extra cost. Therefore other objective will be to minimize the number of shuttles used to complete the loads.

One way to minimize the total number of transfers is to use classes of equivalent products. A product code belongs to a single product class, and it can be replaced by any other product code of the same product class. In this case, if there is not enough stock of a certain code to complete the load, the remaining amount of product can be substituted by other code belonging to the same class and stored in the same depot.

The approach consists in making a plan based in product classes instead of product codes, assigning the appropriate code to the orders at the end of the planning process.

The load capacity of each depot is limited by the number of different bays than can be used simultaneously. This number is not constant and depends on the available staff in each work shift. Even the number of shifts and its duration can change during the week for each depot.

Using Object-Oriented Modelling Technique, we can distinguish the following classes: the DEPOT class with attributes like number of bays, stocks, working hours and shuttle capacity; the LOAD class whose attributes are number of units of each product, number of units of each equivalent class, loading window and duration. Another important class is the WORK PATTERN consisting in a set of time intervals describing the shifts.

At this point it is not difficult to match the problem with the Ilog Schedule classes, Activities and Resources. The LOAD class corresponds almost exactly to an Interval Activity which consumes some resources (stocks) and requires others (bays). On the other hand, DEPOT can not be represented as a single Schedule class. It has to be modelled using several Resources.

For example, the bays are modelled as a Discrete Resource. One bay is required by one load activity and it can be used again once the previous load has finished. The time dependency described by WORK PATTERN is modelled using a resource as a TimeTable in discrete time units. Stocks are modelled as Discrete Resources (one per product) that are consumed by loads, because stocks will not be returned to the depot when the load activity finishes.

It may appear that Schedule will do almost all the work for us. But scheduling is only half of the problem; the other is to assign the right depot to achieve the desired result. We have only mentioned the stock restrictions (combined with the shuttle facility) in making the depot assignment. But the actual problem has some extra conditions: all depots are not indistinguishable and they have some peculiarities. Each one introduces some restrictions that have to be taken into account.

We distinguish between two type of restrictions: the first one, that we have called hard restriction, tells us whether a specific load is able to be loaded in some depot or not. The second type refers to preferences instead of obligations; we have called them soft restrictions. Hard restrictions are easily modelled using Solver variables and creating their corresponding Solver Restrictions. Soft restrictions cannot be implemented using a Solver facility directly. Following analysis is going to give an idea of how soft restrictions become an important part of the problem.

One of the storehouses is associated with the production plant and its load capacity is wanted to be maximized. The plant also produces some goods that should be loaded directly into the trucks instead of being stored (in order to reduce storage cost). This privileged depot is called the plant depot and introduces two different stock types:

Physical Stock is the stock already stored in the depot, and Future Stock is the stock that is being produced during the day. It also introduces a loading priority: the Future stock should be taken if the service quality can be achieved. This model is extended to the other depots since some of them can receive imported goods, and the results are similar to having a production plant in each depot.

The plant depot has a small number of workers. For this reason the company prefers to employ them for loads that need less manipulation time in order to maximize the total number of product units loaded at this location (a whole truck of a single product is easier to load than another loaded with small amounts of a large variety of products). This preference introduces new attribute to the LOAD class, that we called Complexity Level. It is computed based on the manipulation cost of the load. Each depot has its preferred complexity level under which the load will be preferentially accepted.

Putting everything together, we are able to make a model based on Ilog Solver. The last step needed to complete the project is to build a goal for finding the solution. Such goal has to be oriented to reach the desired project objectives. Remember that some objectives are maximizing some criteria, such as the service quality and the load capacity of the plan depot. Other objective is minimizing the total number of shuttles and it can compete with the others, especially if the stock is not evenly distributed.

To centre the problem complexity, OLS has to manage several depots (actually three), about three thousand different product codes, eight hundred equivalent product classes, and it has to make an appointment for about two hundred trucks. The plan has to cover from two to five working days with a discrete time unit of five minutes.

As well as problem requirements, there is a very restrictive operative requisite: the loading plan has to be ready in less than ten minutes of computation time.

We tested the use of a combination of CtMaximize and CtMinimize Solver goals, but even in the best case, the time consumed to reach the solution is too high to make it operative. In some cases, the direct implementation of maximize goal could return CFail (after hours of computations) because these goals are built either to find the optimal solution or give a fail condition if there is no solution.

A typical situation of this kind is present when several loads has to be done at the same time and there are not enough bays in all storehouses to perform the plan. There is an obviously requirement not commented: OLS must make a loading plan. Even in the case described, service quality may be relaxed because delivering with some delay is better than not deliver at all. If the problem has not solution what satisfies all hard restrictions, the goal used should reach the CFail condition as quickly as possible, then some restriction have to be relaxed and try again.

The built goal is based in activity selection criteria, oriented to get a solution close enough to the optimal one. The selector also includes those soft restrictions than can be modelled as a trail order using CtInstantiate goals. To avoid CFail problems, like

described before, new dummy resources are introduced in the model.

OLS finds the solution in two main phases to complete the whole loading plan:

The first one, called assignment phase, assigns loading point, schedules all load and plans the shuttle needs. This phase is compound by several steps. Each one uses the same goal oriented to find a solution that satisfies all criteria. After each step some restrictions are relaxed, if needed, in order to increase the service quality in the next try. As described before, stocks are consumed using the equivalent product class approach.

The final phase, called product code assignment, uses another Solver goal assigning concrete product code of the corresponding equivalent product class. This phase also assigns the desired stock type (physical or future) based in the stock availability and loading time.