# Designing a Generic Human-Machine Framework for Real-time Supply Chain Planning

Jonathan Gaudreault[a], Claude-Guy Quimper[a], Philippe Marier[a*], Mathieu Bouchard[a], François Chéné[a], Jean Bouchard[a]

[a] FORAC Research Consortium, Université Laval
Québec, Canada, G1V 0A6

[*]Corresponding author: philippe.marier@forac.ulaval.ca

*Abstract* — Mixed-Initiative-Systems (MIS) are hybrid decision-making systems in which human and machine collaborate in order to produce a solution. This paper described an MIS adapted to business optimization problems. These problems can usually be solved in less than an hour as they show a linear structure. However, this delay is unacceptable for iterative and interactive decision-making contexts where users need to provide their input. Therefore, we propose a system providing the decision-makers with a convex hull of optimal solutions that minimize/maximize the variables of interest. The users can interactively modify the value of a variable and the system is able to recompute a new optimal solution in a few milliseconds. Four real-time reoptimization methods are described and evaluated. We also propose an improvement to this basic scheme in order to allow a user to explore near-optimal solutions as well. Examples showing real case of how we have exploited this framework within interactive decision support software are given.

***Keywords*** — Linear optimization; Mixed-initiative systems; Supply chain optimization; Human-machine interaction; Tactical planning.

## 1. Introduction

Most decision-making systems (e.g. planning or scheduling systems) found in enterprises lie on one of the following paradigms. The first one is fully automated systems. It is typically the case when an algorithm is used to find an optimal solution to the decision problem. In other cases, the planning is done by a human expert, sometimes with the help of a visual interface allowing him to get real time feedback regarding his decisions and choices. Surprisingly, quite a few optimization problems are planned manually as such. Indeed, automated planning tools are lacking of a political sensitivity or, more generally, do not take into consideration many important soft constraints that are often quite difficult to model (constraints that even human does not realize they exists before he sees a solution violating them).

Mixed-Initiative- Systems (MIS) [1, 2] are hybrid decision-making systems in which human and machines collaborate in order to produce a solution. Most MIS-related research is done by the A.I. community and applies to discrete combinatorial optimization problems. The goal of our research is to propose MIS methods adapted to business optimization problems showing a linear structure, like mid-term/long-term production planning problems. Our approach may be suitable to many kind of industry as it is more problem oriented than industry oriented. It is particularly well suited for problems where human are involved in making a final decision (fine-tuning of the solution by a human is required, or human want to perform what-if analysis). In both situation real-time support by the system (instead of hour-long reoptimization) is needed.

Preliminary notions regarding MIS and supply chain optimization are provided in Section 2. The proposed Mixed-Initiative system for linear optimization is described and evaluated for a real-size industrial supply chain problem in Section 3. Section 4 introduces optimality tolerance for an improved flexibility in exploring solutions. Finally, Section 5 shows how we have exploited this framework within decision support software and presents recent industrial applications. Section 6 concludes the paper.

## 2. Preliminary Notions

### *2.1. Mixed-Initiative Systems (MIS)*

The motivation behind MIS is that human and machines show different strength [3]. Human has an implicit knowledge of the problem that cannot always be formalized. Human guidance can improve the performance of search algorithms [4]. Moreover, the decision-maker is often unaware of a constraint or an issue till he sees it in the solution proposed by the machine. In this context, involving the human in the search for the solution has some values.

The idea of involving human in model optimization can be traced back to as early as 1971 with the work of Benayoun et al. [5] in the context of multiobjective optimization. As mentioned by some authors [6, 7], it is now well accepted that man-machine interaction can be valuable in solving complex optimization problems. Depending on the context, MIS provide different benefits. In some situations, the solution is produced using less computation time because the user can guide the search according to his intuition [4]. This is because humans generally outperform computers in visual perception and strategic thinking. In other contexts, the main interest is for the final solution to get a better acceptance level by decision-makers because it is more in line with informal objectives of the company/decision maker. It is also easier for decision makers to justify and improve solutions in which they participate, and they can implement their preferences and knowledge of the real world without complex and sometimes near-impossible mathematical modeling. Although in theory it is possible to increase user confidence in a solution through the use of explanation functions, building such functions is a difficult task for the designer of a decision-support system [8].

Meignan et al. [6] present an extensive review and a classification of interactive optimization methods in operations research. Their review allowed them to enumerate five classes of interactive optimization approaches. The authors explain that *interactive reoptimization* aims at adjusting the global solution when local changes are made by the user to a given solution. As all constraints are still enforced, local changes are propagated to the global solution through reoptimization [9]. Some interactive reoptimization methods are reported in [10]. In these cases, a reoptimization procedure is applied after a modification of the solution by the decision maker. This is done by freezing into the problem the modifications made by the user before optimizing again. This type of approach is not suitable when the optimization itself takes a long time. Moreover, adding constraints time and time again while interactively finding a satisfying solution can remove a lot of liberty for the solver to find solutions with good objective value. On the other hand, only fixing the last modifications made cannot ensure all previous changes made will be respected. Among examples, Williams et al. [11] are using game-based experiments to assess the potential of "human in the loop" optimization in the context of debris collection problems. In their approach, the user can adjust the multiple objectives themselves and tools are available to create a solution and give instant feedback on the solution they are building.

Most MIS- related researches target discrete combinatorial optimization problems such as timetabling [12], space mission planning and scheduling [13, 14, 15], air traffic control [16], military applications [17, 18] etc. Different application domains present unique challenges. One concern is the diversity of the planning and scheduling constraints that appears in many domains. Smith et al. [19] have developed a computerized framework which allows building mixed initiative systems tailored to specific domain requirements. Their system was primarily used in the planning and scheduling of force deployment. Mixed-Initiative Systems have not yet made their way into business management systems such as those used for Supply Chain Management (SCM). We believe the main reasons are that (1) most of these situations can be modeled as linear optimization problems for which good algorithms exists, and (2) methods developed for classical A.I. planning could not easily be applied to those kind of problems.

## *2.2. Supply chain optimization*

The goal of our research is to propose MIS methods adapted to business problems showing a linear structure model, such as supply chain optimization problems. Supply chains are formed by a set of business units that collaborate to produce goods. *Tactical supply-chain planning* consists of computing the amount of products to produced, consumed, stored or transported for each period of a given time frame with a precise objective to optimize. Common objectives are costs minimization or profit maximization [20, 21]. Tactical supply-chain planning is a very important concern in the forest-products industry [22]. A single business unit produces many products at the same time from a single piece of raw material. This leads to an important interdependency between business units (e.g. forest operations supply many different types of industries at the same time, a sawmill supplies lumber to remanufacturing plants as well as chips to paper mills, etc.). Many mathematical models have been proposed [23, 24] but most of the time they do not exploit decision makers' intuition and preferences.

These problems are typically solved using commercially available software like IBM ILOG CPLEX. They can solve problems having several hundreds of thousands variables in a few minutes, thanks to well-known algorithms like Dantzig's *simplex* method [25].

One shortcoming of these solvers is that they usually return one and only one solution for a given problem (defined as a set of variables, constraints and an objective function) – although thousand of alternative optimal solutions may exist. Most of the time, the returned solution is inadequate as the problem specification does not take into account the contextual and political information known only to the decision maker [4]. It is indeed very difficult in a mathematical model to consider all the preferences of the user. Moreover, the decision maker does not know exactly all the constraints; he "discovers" some when he sees solutions violating them.

Although the solution produced by the solver often contains hundreds of thousands of variables, in practice the decision-maker analyzes the solution by consulting only a few charts, each one containing a few dozens of variables (e.g. 52 variables matching the 52 weeks in the year). An example of this type of chart is provided in Figure 1.

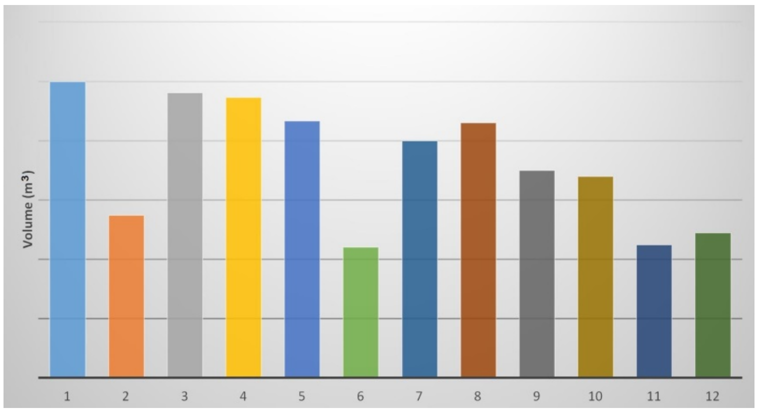
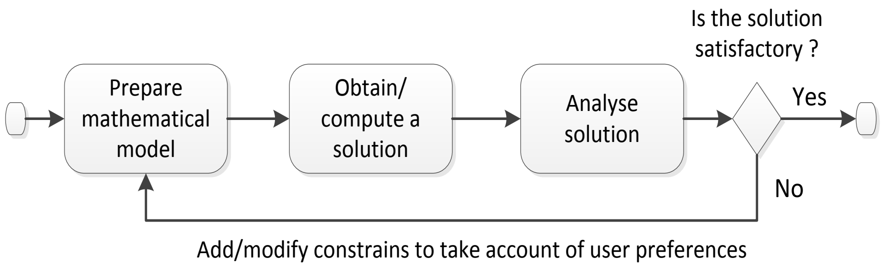


Figure 1. Example of a chart used by decision-makers to analyze the solution of a supply chain optimization problem. This chart could be for example the quantity of a product family to be delivered in a given market over the next 12 months.

When the solution does not satisfy the requirements of the decision maker (e.g. he does not like the value for a specific variable), the mathematical model has to be modified and reoptimized so that a new solution can be found (Figure 2).



Is the solution satisfactory ?
Prepare mathematical model
Obtain/ compute a solution
Analyse solution
Yes
No
Add/modify constrains to take account of user preferences

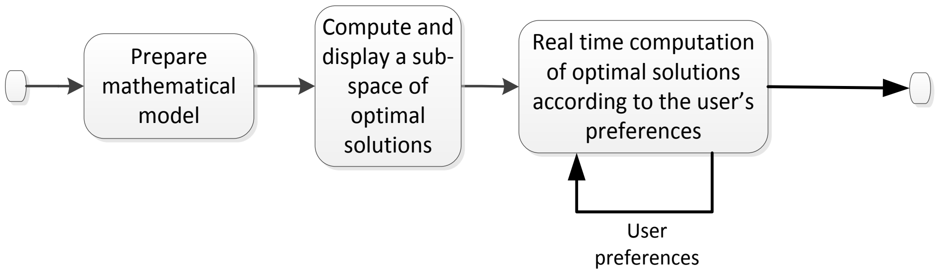
Figure 2. Typical process of finding a suitable solution

Not only is this process very long (solving an integrated supply chain planning problem may easily take up to 60 minutes), this is also a frustrating iterative process because when the decision-maker is not satisfied by a variable's value, he never knows if the variable is really constrained to that specific value in order for the solution to be optimal, or if exists other optimal solutions with other values for that variable. He does not have the choice but to run new optimizations.

## 3. Proposed Mixed-Initiative Framework for Linear Optimization Problems

A good Mixed-Initiative System for linear optimization would provides the user with information regarding whether or not the *variables of interest* are constrained to a certain value in order for the solution to be optimal. It also allows the user to interactively modify the value of a variable and see, in real-time, how the other variables should be modified in response to that modification.

The system we propose allows the user to visualize an implicit sub-space of optimal solutions for a given set of variables. The user can interactively increase or decrease the value of a variable and the system reacts by computing and displaying, in real time, the new sub-space of optimal solutions (Figure 3).



Prepare mathematical model
Compute and display a sub-space of optimal solutions
Real time computation of optimal solutions according to the user's preferences
User preferences

Figure 3. MIS concept applied to linear programming

The variables of interest may vary from decision maker to decision maker. This must be determined with the user for which the decision support system is setup. These variables are typically aggregation variables (ex: quarterly sales by market; production line usage per month). Good representation of these variables (in Excel charts for example) must be set such that it is manageable from a human point of view.

## *3.1. Obtaining and displaying a sub-space of optimal solutions*

To palliate the fact that classical solvers return a single solution, we proceed as follows. We use a classical solver to compute an optimal solution for the problem *P*. Once the optimal objective value is known, we search for other optimal solutions. We create a new instance *P'* by augmenting the instance *P* with a constraint forcing the objective value to be equal to the optimal value found for *P*. The space of *feasible* solutions of *P'* is therefore equivalent to the space of *optimal* solutions of *P*. For each of the *n* variables $x_i$ displayed on the graph, we search for a solution $\underline{s_i}$ that minimizes $x_i$ and a solution $\overline{s_i}$ that maximizes $x_i$ while preserving the optimality of the solutions. These $2n$ computations can be performed in parallel using a supercomputer[1]

Then, one can display on the chart (see Figure 4) the *range of optimality* for each variable $x_i$ (the minimum and maximum values the variable can take in an optimal solution).

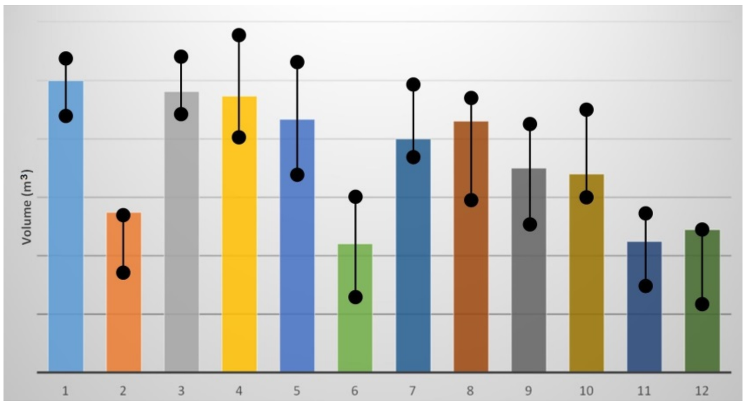


Figure 4. An optimal solution with range of optimality for the variables.

Moreover, it is possible to construct an "average" solution to be displayed to the user by summing the $2n$ solution vectors and dividing the result by the scalar $2n$. The validity of this process relies on the concept of convexity that we present bellow.

A *n*-dimensional vector represents the values assigned to *n* variables, and therefore a solution to a problem. A vector also represents a point in an *n*-dimensional space. Let *x* and *y* be two *n*-dimensional vectors. A linear combination is the weighted sum of two or more vectors, e.g. $\alpha x + \beta y$. A linear combination is affine if the sum of the weights equals one, e.g. $\alpha x + \left(1-\alpha\right) y$. Any point on the line passing by the points *x* and *y* is an affine combination of *x* and *y*. A linear combination is convex if it is affine and if each weight is non-negative, e.g. $\alpha x + \left(1-\alpha\right) y$ for $\alpha \in \left[0,1\right]$. The points that can be obtained by a convex combination of *x* and *y* are those on the segment connecting *x* to *y*. A space *S* is convex if any convex combination of two points in *S* is also in *S*.

Theorem 1 is a well-known result in the linear programming literature (e.g. [26]). It shows how an optimal solution to a linear program can be obtained from the convex combination of distinct optimal solutions. We reproduce the proof of the theorem as it shows the properties we will exploit in our system.

**Theorem 1 : The convex combination of two optimal solutions of a linear program is also an optimal solution.**

The space $S = \left\{x \,\middle|\, Ax \le b,\ x \ge 0\right\}$ contains the feasible solutions of a linear program. Let $x \in S$ and $y \in S$ be two solutions. We show that any convex combination of *x* and *y* belongs to *S*.

$$\begin{aligned} A(\alpha x + (1 - \alpha)\, y) &= \alpha Ax + (1 - \alpha)\, Ay \\ &\le \alpha b + (1 - \alpha)\, b \qquad \text{since } \alpha \in [0, 1] \\ &= b \end{aligned}$$

Suppose that *x* and *y* are two optimal solutions to the linear program, i.e. *x* maximizes the scalar product $c^{\mathrm{T}} x$ and so does *y*. Let $c^* = c^{\mathrm{T}} x = c^{\mathrm{T}} y$ be the optimal cost. We show that any convex combination of *x* and *y* is also an optimal solution. □

The convex hull of a set of points $X = \left\{\vec{x}_1, \ldots, \vec{x}_n\right\}$ is the space containing all convex combinations of the points in *X*. Formally, we note

[1] If parallel computing is not available, a single computer can run overnight prior to using the interactive tool.

$$\operatorname{conv}(\vec{x}_1,\ldots,\vec{x}_n) = \left\{ \sum_{i=1}^{n} \alpha_i \vec{x}_i \,\middle|\, \alpha_i \geq 0, \sum_{i=1}^{n} \alpha_i = 1 \right\}$$

From Theorem 1, if the points $\vec{x}_1,\ldots,\vec{x}_n$ are optimal feasible solutions to a linear program, then all points in $\operatorname{conv}(\vec{x}_1,\ldots,\vec{x}_n)$ are optimal feasible solutions.

In our system, the convex hull of the solutions $\underline{s_i}$ and $\overline{s_i}$ that minimize and maximize each of $n$ variables form a subspace of optimal solutions.

### 3.2. *Real-time convex combination of optimal solutions*

A chart like the one depicted in Figure 4 allows the user to see the flexibility (range of optimality) he has for any of the variable of interest. He can thus change the value of a variable within the optimality range of that variable, for example by dragging up or down the top of a bar in the chart. The system should then adjust the value taken by the other variables such that the solution is still optimal. Normally, this operation would require a lengthy complete re-optimization of the problem. Based on Theorem 1, it is possible to compute a new optimal solution by generating new convex combination of the optimal solutions obtained from the previous section. To find a solution with a given variable assigned to a specific value, it is sufficient to compute a new convex combination of extreme solutions. If this could be done in real-time, we could instantly refresh the chart and display the impact of the user modification on the other variables. The user could thus *navigate* through the solutions space, successively modifying several variables until a suitable solution is found.

Figure 5 illustrate this idea for a small problem with two variables. The polygon shows the set of convex combination formed by optimal solutions minimizing and optimizing each of the variables[2].

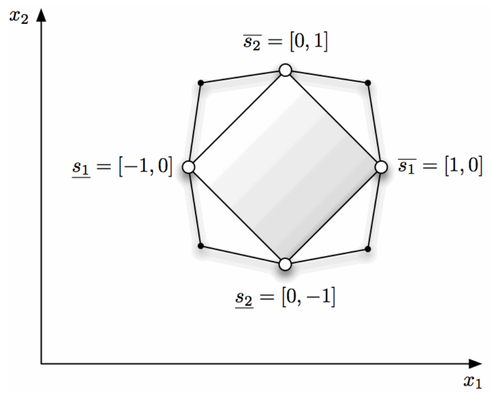


Figure 5. Available optimal zone using convex combinations.

When the user modifies a variable, we do not just want to find any point in the sub-space such that the modified variable takes the desired value; we want to find a solution such that the other variables move as little as possible so that the overall system appears to be *stable*. If it would not be the case, then any modification made to a variable could offset the previous modifications made to the other variables and it would never be possible for the user to converge to a satisfactory solution.

Given a solution $x$, a variable $x_i$, and a value $v$, finding a new solution $x'$ where $x'_i = v \neq x_i$ while changing as little as possible the values of the other variables is an optimization problem.

We propose four approaches to solve this optimization problem. Each of them is a trade-off between *responsiveness* and *stability*.

**Definition 1:** *Responsiveness* refers to the computation time needed to obtain a new solution each time the decision-maker modifies a variable. It determines the time before the software can display the new optimized solution.

**Definition 2:** *Stability* is the property of computing a new solution that is as close as possible from the current solution.

While responsiveness guarantees that the software reacts in real-time to the changes of the decision maker, stability ensures that the software does not move the solution away from the choices that the decision maker previously made.

Responsiveness is measured in milliseconds. Stability is given by the Euclidean distance between the original solution $x$ and the modified solution $x'$.

[2] This example (Figure 5) also shows that there could be optimal solutions not covered by this sub-space. This is why we say that the user navigate in a space (formed by convex combination of our extreme solutions) that is in reality a sub-space of the optimal solutions.

Since responsiveness is an important criterion, we exclude the method of resolving the original problem from scratch. We restrict ourselves to find a solution in the convex hull of the $2n$ precomputed solutions $\underline{s}_1, \overline{s}_1, \ldots, \underline{s}_n, \overline{s}_n$, a much smaller problem then the original one. Here are the proposed methods.

**1. Minimizing the Euclidean distance.** This method aims at finding in the convex hull $\operatorname{conv}(\underline{s}_1, \overline{s}_1, \ldots, \underline{s}_n, \overline{s}_n)$ the solution whose Euclidean distance is the closest to $x$. Let $S$ be the $n \times 2n$ matrix whose columns are the solutions $\underline{s}_1, \overline{s}_1, \ldots, \underline{s}_n, \overline{s}_n$. The solution we are looking for is a convex combination of these solutions. We are therefore looking for a vector $\alpha$ such that $x' = S\alpha$, $\sum_{i=1}^{2n} \alpha_i = 1$, and $\alpha_i \geq 0$. Moreover, we want $x'_i$ to be equal to $v$. We obtain the following quadratic problem where $e_i$ is the vector with null components except for the $i^{\text{th}}$ one that is equal to one, $\vec{1}$ is the vector with all components equal to one, and $\vec{0}$ is the null vector:

$$
\begin{array}{lll}
\min & \| x - S\alpha \|^2 & \\
\text{s.t.} & e_i^T S\alpha & = v \\
 & \vec{1}^T \alpha & = 1 \\
 & \alpha & \geq \vec{0}
\end{array}
$$

The squared Euclidean distance $\|x - S\alpha\|^2$ can be rewritten as $\alpha^T S^T S\alpha - 2x^T S\alpha + x^T x$. The problem is a semidefinite program. The objective function is quadratic and convex and the constraints are linear. Solvers like CPLEX can solve this problem using an interior point algorithm.

**2. Minimizing the maximum distance.** Even though semidefinite problems can be efficiently solved, they are more computationally demanding than linear programs. We propose another approach that is likely to be faster. Rather than minimizing the Euclidean distance between the new and the former solution, we minimize the maximum distance between two variables, i.e. we minimize $g = \max_j |x_j - x'_j|$. The linear program we present is inspired from the previous one. Once more, we compute the vector $\alpha$ that expresses $x'$ in terms of a convex combination of the $2n$ solutions stored in the columns of $S$ ($x' = S\alpha$). The two first constraints of this linear program force the variable $g$ to be at least as large as $|x'_j - x_j|$. The unknown of this problem are the vector $\alpha$ and the gap variable $g$.

$$
\begin{array}{lll}
\min & & g \\
\text{s.t.} & S\alpha - \vec{1}g & \leq x \\
 & -S\alpha - \vec{1}g & \leq -x \\
 & e_i^T S\alpha & = v \\
 & \vec{1}^T \alpha & = 1 \\
 & \alpha & \geq \vec{0}
\end{array}
$$

This problem can be solved using the simplex method, which is usually faster than the interior point method used for semidefinite problems of the previous approach. The obvious drawback is that the objective function is not completely in line with our stability metric. Even if we minimize the distance between the two variables that are the furthest apart, we do not minimize the distance between the other variables. A solution $x'$ where all variables are changed by the same amount would be equivalent to a solution where only the variable $x_i$ is modified.

**3. A bipolar heuristic.** We introduce a third method that focuses on responsiveness. The *bipolar heuristic* is a very fast way to compute a new solution. Given that the decision maker wants to set the variable $x_i$ to the value $v$, we compute the unique convex combination $x' = \alpha \underline{s_i} + (1 - \alpha)\overline{s_i}$ such that $x'_i = v$. To do so, we set $\alpha\ (\overline{s_{ii}} - v)/(\overline{s_{ii}} - \underline{s_{ii}})$ where $\underline{s_{ii}}$ and $\overline{s_{ii}}$ are the values of $x_i$ in the precomputed solutions that minimize and maximize $x_i$. The geometric interpretation of this method is that we choose $x'$ to be the intersection of the segment connecting $\underline{s_i}$ to $\overline{s_i}$ and the plane $x'_i = v$.

This solution is by far the most responsive one, as it does not involve any solver. However, it does not take into account the stability metric. Indeed, a small change on the variable $x_i$ can produce big changes on the other variables of the solution. For example, consider an example in the plane where we have $\underline{s}_1 = [-1, 0]$, $\overline{s}_1 = [1, 0]$, $\underline{s}_2 = [0, -1]$, and $\overline{s}_2 = [0, 1]$ (see Figure 6). Let the current solution be $x = [0.5,\ 0]$. A small move of variable $x_2$ from 0 to $\varepsilon$ causes the solution to change for $x' = [0,\ \varepsilon]$ even though the solution $x' = [0.5,\ \varepsilon]$ is a better choice.

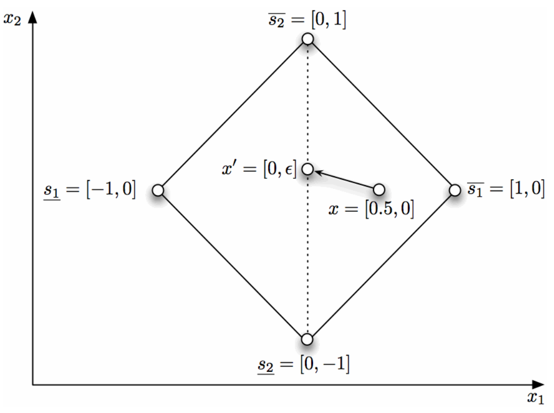


Figure 6. Illustrating the bipolar method stability drawback.

**4. The triangular heuristic.** To palliate to the non-stability of the previous approach, we propose the *triangular heuristic*. First, this method determines whether the variable $x_i$ is increased ($v > x_i$) or decreased ($v < x_i$). If the variable is increased, we compute the unique convex combination between the current solution $x$ and $\overline{s_i}$ that intersects the plane $x_i = v$. If the variable is decreased, we compute the unique convex combination between the current solution $x$ and $\underline{s_i}$ that intersects the plane $x_i = v$. In other words, if the decision-maker increases $x_i$, we set

$$x' = \alpha x + (1 - \alpha)\overline{s_i}$$

where

$$\alpha = \frac{\overline{s_{ii}} - v}{\overline{s_{ii}} - x_i}$$

If the decision maker decreases the variable $x_i$, we set

$$x' = \alpha x + (1 - \alpha)\underline{s_i}$$

where

$$\alpha = \frac{\underline{s_{ii}} - v}{\underline{s_{ii}} - x_i}$$

Going back to the example of Figure 6 where $x = [0.5, 0]$. Increasing $x_2$ by $\varepsilon$ sets the new solution to $x' = [0.5 - \varepsilon/2,\ \varepsilon]$. This is significantly more stable that the bipolar heuristic.

There is a second difference between the bipolar heuristic and the triangular heuristic. The bipolar heuristic can only produce a solution that lies on a segment connecting a solution $\overline{s_i}$ to a solution $\underline{s_i}$ for a given $j$. The triangular heuristic can produce any solution in the convex hull $\text{conv}(\underline{s_1}, \overline{s_1}, \text{K}\ , \underline{s_n}, \overline{s_n})$. Indeed, there exists a sequence of movements that the decision maker can make in order to reach any solution in the convex space.

**Property 1:** A sequence of movements made with the triangular heuristic from a solution $x$ can reach any solution in the convex hull $\text{conv}(\underline{s_1}, \overline{s_1}, \text{K}\ , \underline{s_n}, \overline{s_n})$.

*Proof.* Let $x'$ be the final solution in $\text{conv}(\underline{s_1}, \overline{s_1}, \text{K}\ , \underline{s_n}, \overline{s_n})$. Let $S$ be the matrix whose columns are the extreme points of the convex hull. There exists a non-negative vector $\alpha$ whose components sum to one and that satisfies $x' = S\alpha$. We prove by induction on the number of non-null components in $\alpha$ that any $x'$ is reachable from a sequence of triangular movements.

Suppose that there exists a single non-null component in $\alpha$. This component is equal to one and we have $x' = \overline{s_i}$ or $x' = \underline{s_i}$ for some $j$. Increasing the value of $x_j$ in the current solution to its maximum or minimum makes the triangular heuristic fix the new solution to $x'$.

Suppose that one can reach any solution $x = S\alpha$ with $k > 0$ non-null components in $\alpha$, we prove that one can also reach any solution $x' = S\alpha$ with $k + 1$ non-null components in $\beta$. Let $x' = S\beta$ be a solution with $k + 1$ non-null components in $\beta$. Let $j$ be the index of any non-null component in $\beta$. We construct the vector $\alpha$ as follows.

$$a_i = \begin{cases} 0 & \text{if } i = j \\ \dfrac{\beta_i}{1 - \beta_j} & \text{if } i \neq j \end{cases}$$

The solution $x = S\alpha$ is reachable since it has only $k$ non-null components. Moreover, let $s \in \{\underline{s_k}, \overline{s_k}\}$ be the solution associated to the component $\beta_j$. It is possible to move from the solution $x$ to $x'$ using the triangular heuristic since this relation holds.

$$x' = (1 - \beta_j)x + \beta_j s = S\beta$$

This is the relation computed when the decision maker changes the value of $x_k$ to $x'_k$. □

Property 1 ensures that any solution in the convex hull can be reached.

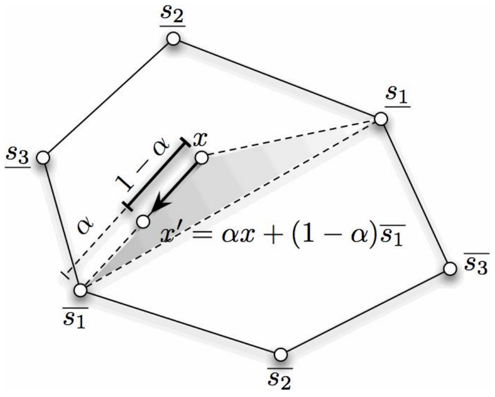


Figure 7. Illustrating the triangular method.

## 3.3. *Evaluation of the proposed approaches*

The industrial partner for the following case study was looking to put in place an efficient process for integrated sales and operation planning (S&OP). As part of this process is the use of a large scale linear programming model with over 200,000 variables and 100,000 constraints introduced in [21]. This model generates a unified tactical plan for the production and distribution network. This plan is divided in fifty two weekly periods and integrates decisions pertaining to production, distribution and sales.

### 3.3.1. *Industrial case*

**Production**. Lumber production from sawmills involves three main processes: sawing, drying and planing. The *sawing* process is divergent as a single product entering the production line gets transformed into several different products. Different production recipe can be chosen in order to influence the products' basket that gets produced. Lumber *drying* is done by batch in large kilns and the process can lasts from 2 to 5 days depending on several factors like the species, lumber dimension and original moisture content. Kiln drying may be preceded by an air drying operation that can lasts from 2 to 18 weeks. Finally, the *planing* is the process that gives the lumber its finish and exact dimension. Because of the wood characteristics and defects, not two lumber units entering the planing process are exactly identical. It is therefore impossible to precisely predict the final products exiting the planing process given a set of input products. Historical production planing data are used to estimate the production of a given input product. Tactical production planning helps determining, for each sawmill in the network and for each week, the sawing recipe to use, the volume of each sawed product to air dry, the volume of each product to kiln dry and the volume of each product to plane along with the planing recipe to use.

**Sales**. In the North American forest products industry, lumber market prices follow a seasonal pattern. The tactical planning model takes this into account and may suggest producing in advance some quantities of a given product that will be sold at a later time during the year when the market price is more profitable. It also considers the limited volume that can be sold in given markets per product per period. The user running the model can also constrains the percentage of the production that goes to service each market.

**Distribution**. The company runs three sawmills in the province of Quebec in Canada and makes use of two distribution centers in the eastern United States and one in Quebec. It is not always required for the lumber products to go through a distribution centre to reach a market, but it is sometimes more economical. Transportation can also occur between mills as in the company' network, not all sawmills have planing capability. From a sawmill with no planing capability, the dry lumber can either be moved to another sawmill with planing capability or it can be transported to a distribution centre or be sent directly to a customer. The model help determine where to send the lumber along with the transportation mode to use which can be either by trucks or by rail.

Planning the overall network of sawmills taking into consideration market requirement, price fluctuation, production capacity and transportation alternatives and costs is not an easy task. To solve to optimality this complete problem can take up to 45 minutes using CPLEX, one of the leading software to solve linear programming models.

### *3.3.2. Experiments*

The main purpose of these experiments was to evaluate the behavior of the system in terms of stability and responsiveness metrics (as defined in Section 3.2) for the proposed approaches.

We first generated an optimal solution for the industrial case introduced previously. Then, we simulated the situation where the decision-maker visualizes various charts with different number of variables on them (1, 10, 20, 30, 40, and 52 variables). For each variable we computed its range of optimality. We then simulated the situation where the decision-maker asks for modification of some variable values. We tested the behavior of the system for small modifications (the variable is increased or decreased by a gap corresponding to 7% of its range of optimality), medium modifications (45% of the range) and large modifications (75% of the range).

After each modification, an optimal solution is computed using one of the four approaches and we measure responsiveness (computation time) and stability (Euclidean distance between original and recomputed solutions).

### *3.3.3. Results*

Figure 8 provides results for responsiveness of the system (in milliseconds). We recall that solving the original problems each time a modification is asked by the user would require a 45 minutes for each modification. Using the proposed approaches the worst case observed was a recomputation time of 600 milliseconds (minimization of the Euclidean Distance approach), see subfigure i. We were not expecting this method to perform so well because of the computation time required to solve a quadratic optimization problem. Nonetheless, it appears the problem is small enough to provide good performance.

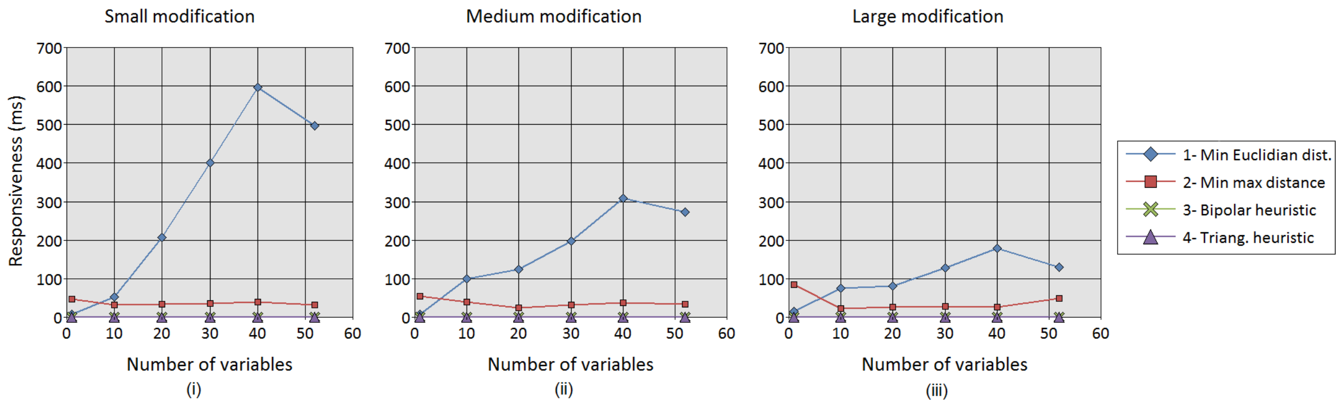


Figure 8. Solution recomputation time (in milliseconds) in reaction to small (i), medium (ii) and large (iii) variable value modifications, according to the number of variables displayed.

However, as expected, computation time rapidly grows with the number of variable displayed on the charts. Clearly, in a situation where the decision maker would be visualizing 4 or 5 charts with 52 variables on each of them, the Euclidean approach would not meet our real-time expectations.

Subfigures i, ii, and iii show that the size of the modification asked by the user has a strong effect on the responsiveness of the Euclidean approach. The larger the modification is (in terms of % of the optimality range) the easier is the computation of the new solution. The explanation is the following. In the extreme case where the user constrains the variable to take its value $\overline{x}_i$ , the system is limited to the solutions corresponding to an extreme point of our convex hull. Said otherwise, there is not much exploration to perform.

The responsiveness of the method that minimizes the maximum distance is not affected by how much a variable is changed. Its linear optimization problem is small and easy to solve – although it is of not as efficient as the heuristic methods. However, those three approaches meat our real-time expectations.

Figure 9 shows the Euclidean distance between the original and the recomputed solution after a small (i), medium (ii) or large (iii) modification and according to the number of variables displayed. The method that minimizes the Euclidean distance always offers the best results.

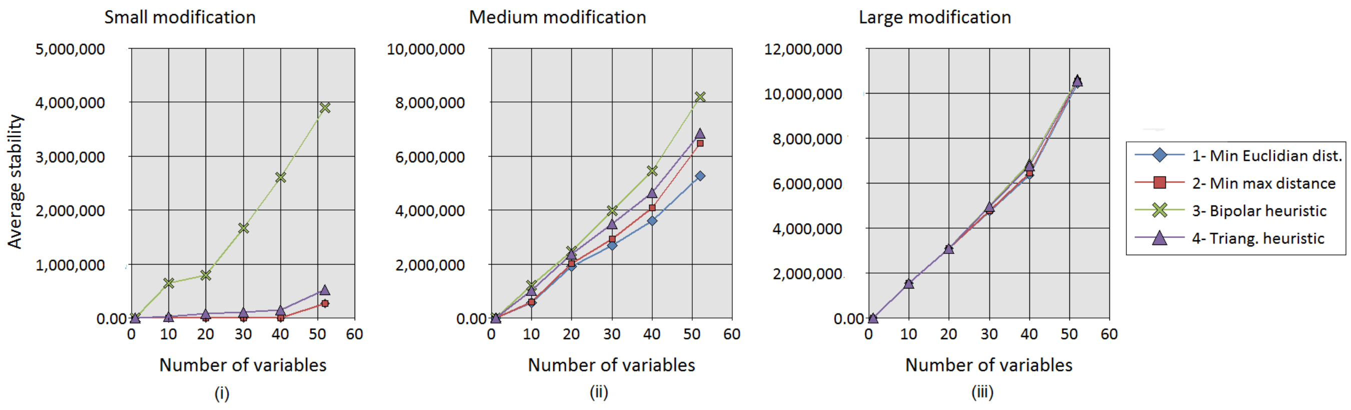


Figure 9. Euclidean distance between the original and recomputed solution after a small (i), medium (ii) or large (iii) variable value modification, according to the number of variables displayed.

As expected from Section 3.2, the bipolar heuristic gives the worst results. Small modifications of a variable value (subfigure i) lead to huge modifications to other variables. Even worse, the beta-testers considered unacceptable that the new solutions are computed without any consideration for changes they made previously.

For small modifications (subfigure i), minimization of the maximum distance or using the triangular heuristic provides stability very close to the Euclidean approach. Actually, for the minimization of the maximum distance, results on the chart are indistinct from those of the Euclidean approach.

However, minimizing the maximum distance sometimes leads to the problematic situation discussed in Section 3.2, i.e. the largest change is minimized but the other variables move for no valid reason. The triangular heuristic does not have this drawback.

We then analyzes the impact of small, medium and large modifications (comparing subfigures i, ii and iii). The larger the modification asked by the user is, the more the different approaches give similar results. As discussed previously, large modifications to a variable value lead to a very small space of optimal solutions to which all the approaches are restricted.

We can summarize this section by saying the minimization of the maximal distance and the bipolar heuristics both have drawbacks in terms of solution stability. The minimization of the Euclidean distance is optimal regarding this aspect. It should be used in situations where the decision-makers based their decision on the analysis of a small subset of variables (typically a chart with a dozen of variables) although the optimization problem can have hundreds of thousands of variables. If the decision-makers need to simultaneously analyze and modify charts with hundreds of variables, the triangular heuristic should be used as our experiments reported near optimal performance.

## 4. Introducing optimality tolerance

The proposed method so far allows exploring optimal solutions. Decision makers may be interested in exploring near-optimal solutions too (as the data used to model the problem are often rough estimates, it would be pointless to exclude solutions that decrease profit by only a few dollars). A *naïve* approach could be the following. When computing solutions minimizing/maximizing a given variable, we tolerate a gap between the new solution's profit and the optimal solution of the original problem. However, this approach would not work very well when using the triangular heuristic. Interaction with the system revealed frequent cases where adjusting a variable to values within the optimal space would return a near-optimal solution while some existing optimal solutions would clearly be a better choice. We aim to explain the cause of this behavior and propose another approach. When computing a new solution with the triangular heuristic, the values of the variables are interpolated between two solutions, so are the profit values associated to these solutions. Therefore, if one solution is optimal but not the other, unless the weight of the optimal solution is 0 and the weight of the non-optimal solution is 1, then the new solution resulting from the combination of these solutions will not be optimal.

This was not an issue with the original triangular method (previous section) where all solutions are optimal. However, if we include optimality tolerance, we combine non-optimal solutions. This result is that the system (1) returns a non-optimal solution as soon as a variable is modified towards an extreme value associated with a non-optimal solution even if the target value is within the optimal range for that variable and (2) once a non-optimal solution is reached, the system does not return an optimal solution unless a variable is set to an extreme value associated with an optimal solution.

This behavior is explained in Figure 10. In this example, the dark grey area is the real optimal solution space. The light grey area shows how this space expands when we tolerate an optimality gap. In this example, we suppose the system first displays the solution for which $y$ is maximized and then the user asks to gradually decrease $x$. As soon as $x$ starts to decrease, the triangular heuristic gets a sub-optimal solution as it interpolates between $\overline{y}$ and $\underline{\underline{x}}$. This is a pity as it is clear that for some values for $x$ we could still have an optimal (dark grey) solution.

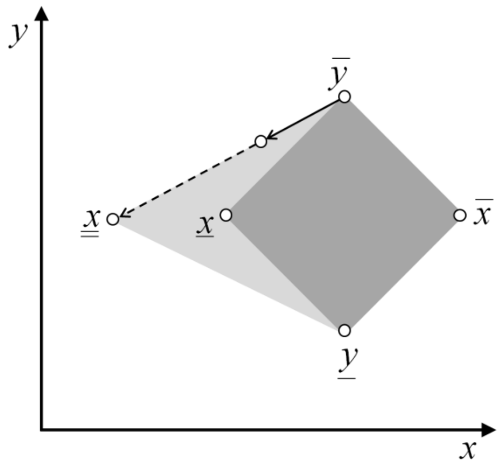


Figure 10. Behavior of original Triangular heuristic when provided with a solution space with optimality gap

The situation is even worse for the following reason. Suppose we use a solver (e.g. the simplex method) to find a solution minimizing/maximizing a given variable, with the constraint that the profit of the network should be at least 95% of the profit of the original optimal solution. The simplex could return a solution minimizing/maximizing the variable that has a profit of 95% even if there exists another solution where the variable takes the exact same value while showing a profit of 100%.

To palliate this situation, we introduce a multi-stage approach in the next subsection.

## *4.1. A multi-stage approach*

We want the system to return an optimal solution whenever possible. To achieve this, we need two pairs of solutions for each variable $x_i$ the user is interested to modify. As in original method, one pair of solutions $\left\{\underline{x_i}, \overline{x_i}\right\}$ minimizing/maximizing the variable value while preserving optimality and another pair $\left\{\underline{\underline{x_i}}, \overline{\overline{x_i}}\right\}$ minimizing/maximizing the variable value obtained when the optimality constraint is relaxed (according to some gap specified by the user). We also modified the triangular heuristic to interpolate using tolerated extreme points $\underline{\underline{x_i}}$ and $\overline{\overline{x_i}}$ only when necessary.

This behavior is illustrated in Figure 11. In this example, we suppose the system first displays the solution for which $y$ is maximized. Then the user asks to gradually decrease $x$. As long as $x$ is greater than the value it has in solution $\underline{x}$, we interpolate between the solutions $\overline{y}$ and $\underline{x}$ and the solution is still optimal. Past that point, we interpolate between the solutions $\underline{x}$ and $\underline{\underline{x}}$.

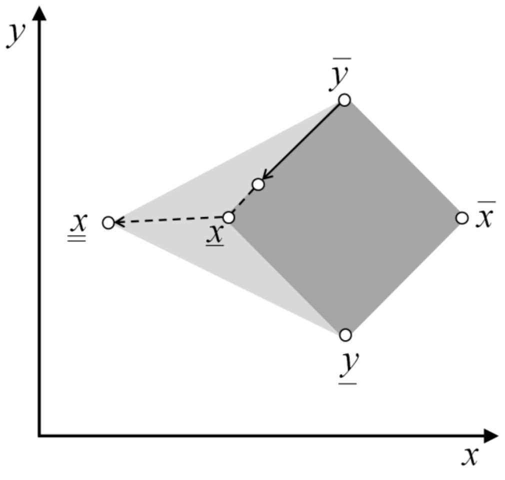


Figure 11. Behavior of the multi-stage approach

## *4.2. Experiments*

We compared the proposed multi-stage approach with the original triangular method and to the triangular method with tolerance.We consider another supply chain optimization problem (wood flow supply chain). This problem is graphically represented in Figure 12. Logs are harvested from a forest and can either be transformed into paper or pellets. Both products cost the same to produce and generate the same income per log. Paper demand is below production capacity, implying that a minimal amount of pellets must be produced to maximize the profit. We suppose a planning horizon of 3 periods and we have a total of 37 variables. One variable is the objective-function "profit" variable (1 variable). Then we have flow variables (3 periods and total) from forest to each of the mills and from each of the mills to their market (16 variables). There is the volume sold (3 periods and total) to each market (8 variables). Finally, we have forest harvesting and production at each of the mills (3 periods and total), which account for 12 variables. In this small case, all of the decision variables (except the one representing the objective function value) are allowed to be changed by the user, but in a supply chain context, the user will typically want to change aggregated flow volume between units in the network and facility utilization.

Using the original *triangular original* approach, the range of optimality of the variables is rather small. The decision maker now wants to interactively explore/discover near optimal solutions (max gap. 5 %) that best meet his preferences. Introducing tolerance should allow decreasing pellet production, and/or reduce forest harvesting volumes.

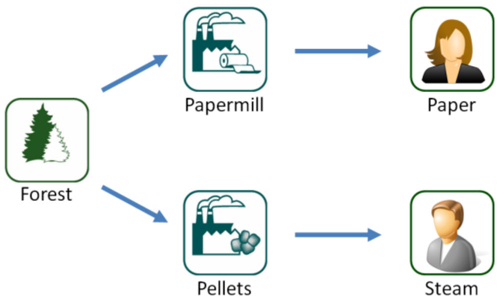


Figure 12. Small wood supply chain

Using the *triangular with gap tolerance* approach, variables should provide a wider range of optimality (within the allowed 5 % gap) but most manipulation of a variable by the user should lead to non-optimal solutions (even though variables are manipulated within their range of optimality).

With the *multi-stage* approach, we should have the same range of optimality for the variables, but a small modification to these variables should lead to optimal solutions. In the following experiment, we formally compare the approaches according to these criteria: (1) range of optimality of the variables, and (2) the optimality of the solutions that are computed after a modification to a variable.

For each method, we computed the range of optimality of each variable. Then, for each variable, we incremented/decremented the variable value and measured the gap between the new solution and the original one. Figure 13 shows how the range of optimality of individual variables are extended (in comparison to the original triangular method) when we allow a 5 % gap. For 16 variables, we see no increase in the range of optimality when using our new approach instead of the original triangular approach. These are mainly the variables linked to the paper portion of the supply chain: product flow from forest to papermill, papermill production, product flow from papermill to paper market, volume of paper sold. We see a 5% increase in the range of optimality for 6 other variables (total of volumes transported in the energy portion of the supply chain; total forest harvested and total profit) and a 15% increase for 15 other variables (single period value of the variables linked to the energy portion of the supply chain and forest harvesting).

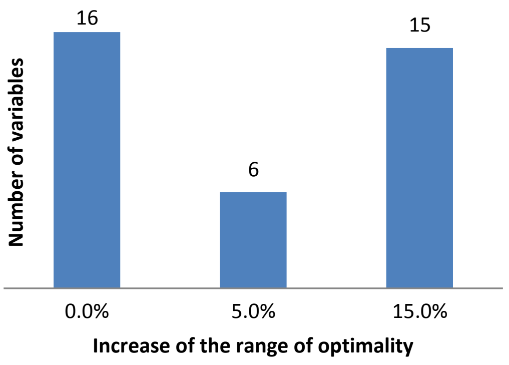


Figure 13. Increase of the range of optimality of the individual variables when provided with a 5% gap on the solution objective function

The ill effects of the *triangular with gap tolerance* approach appeared as expected. Out of the 72 “extreme” solutions, ( $\underline{\underline{x}}$ and $\overline{\overline{x}}$ minimizing/maximizing the 36 manipulable variables), 52 were non-optimal.

Except for the pellet production and the forest harvesting variables, even small modification to a variable leads to a sub-optimal solution (in which pellet production and forest harvesting decrease). Consequently, the system gives the users the false impression they cannot modify these variables without altering optimality.

Using the *multi-stage* approach, out of the 144 solutions corrresponding to $\underline{x}$, $\overline{x}$, $\underline{\underline{x}}$, $\overline{\overline{x}}$, only 20 were sub-optimal (they correspond to the situations where $\underline{\underline{x}} < \underline{x}$ or $\overline{\overline{x}} > \overline{x}$ ). All other manipulations of a variable in the range between $\underline{x}$ and $\overline{x}$ lead to optimal solution. This can be verified for any variable but we report results for one specific variable in Figure 14. For the purpose of our demonstration, we selected a variable that has some range of optimality even when the optimality gap is zero.

The blue curve shows that using the original triangular method, we can modify the variable from $\overline{x}$ to $\underline{x}$ and from $\underline{x}$ to $\overline{x}$ and the new computed solution is always optimal.

Using the triangular method with gap tolerance (in red) we leave the optimality region as soon as we move from $\underline{x}$ toward $\underline{\underline{x}}$.

With the multi-stage approach (in green), the solution is optimal between $\overline{x}$ and $\underline{x}$. We leave optimality only if we go below $\underline{x}$ toward $\underline{\underline{x}}$, which is the expected behavior. A similar behavior is shown when we move back to $\overline{x}$.

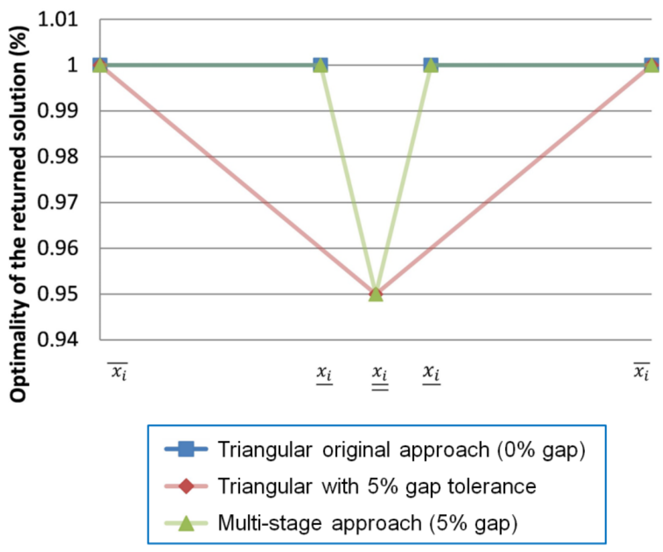
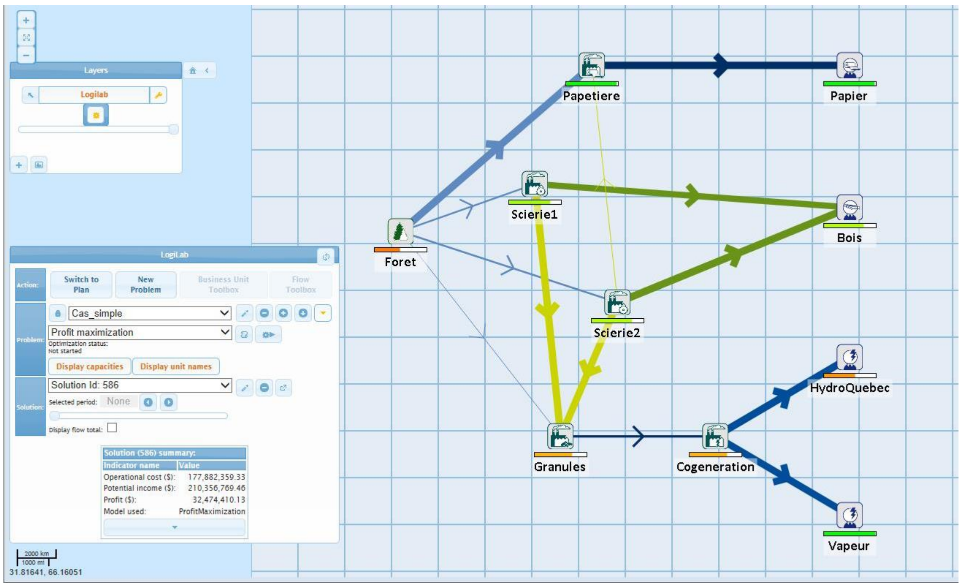


Figure 14. Comparison of optimality of returned solutions during interaction. For this variable, the solutions for $\overline{x}$ and $\overline{\overline{x}}$ were identical, this is why $\overline{\overline{x}}$ does not appear in the chart.

## 5. Exploiting the Framework Within Interactive Decision Support Softwares

### *5.1. Interactive Supply Chain Tactical Optimization in LogiLab*

The FORAC Research Consortium previously developed a software called LogiLab to design, analyse and plan of forest-products supply chains. The mathematical optimization model used by LogiLab is described by Jerbi et al. [23]. Since LogiLab’s data presentation is strongly evocative and natural for planners, it makes a natural basis for the integration of our interactive system.



Cas_simple
Profit maximization
Optimization status:
Not started
Solution Id: 586
Selected period: None
Display flow total:

| Solution (586) summary: | |
|---|---|
| Indicator name | Value |
| Operational cost ($): | 177,882,359.33 |
| Potential income ($): | 210,356,769.46 |
| Profit ($): | 32,474,410.13 |
| Model used: | ProfitMaximization |

Figure 15. LogiLab graphical user interface

In the original LogiLab, the products flows from one business unit to another are presented by arcs connecting the corresponding nodes. The thickness of each arc is proportional to the volume of products that flows from its origin to its destination (see Figure 15). As we now want to display the “range of optimality” of each individual variable, we proposed replacing each arc by a “pipe”. According to this metaphor, the diameter of the pipe represents the maximum value the flow variable can take. The colored part of the pipe represents the current level of goods that flows through the pipe (i.e. value of the variable in the current optimal solution). To represent the minimal value that allows for the solution to be optimal, the bottom part of the flow in the pipe is displayed with a darker color (see Figure 17).

The simplest and most intuitive way to allow the user to interact with the flows (in order for him to adjust the value of a variable) is simply to allow him to click on the pipe and then drag the mouse within the range of optimality. However, most flows tend to be too small to enable proper manipulation. Therefore, when the user rolls the cursor over a flow, an enlarged cross-section appears and allows easier manipulations (see Figure 16). The interaction scenario is as follow: (1) The user rolls over a flow; the zoom bubble (that resembles the cross-section of a pipe) appears and the user is allowed (2) to set a new value for this flow variable. Finally (3), the system updates in real time the value of the other variables, thus computing a feasible solution that is still optimal.

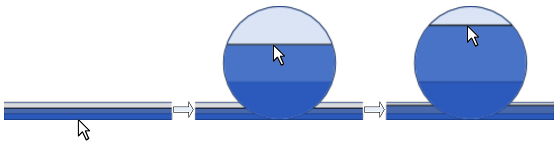


Figure 16. Interaction with flows within the proposed system

In the original LogiLab, the percentage of the total capacity of a business unit used is displayed using a rectangle similar to a progression bar. We added visualization of the range of optimality data for these variables using a color code similar to what we proposed for the flow variables. Figure 17 presents the graphical user interface integrating those concepts.

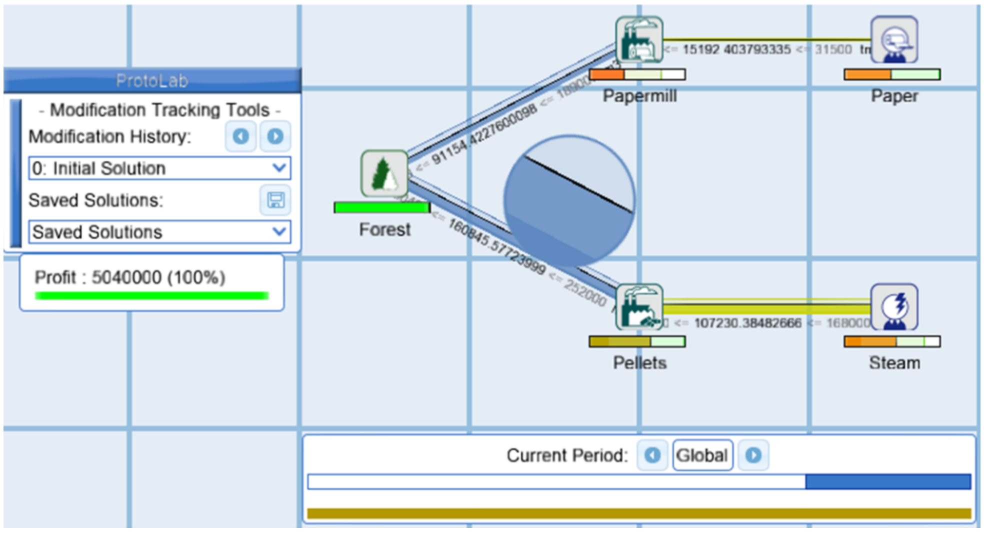


Figure 17. Proposed graphical user interface for interactive supply chain tactical optimization

This real-time interaction with a supply chain solution was shown to our industrial partners who were quite impressed by the possibilities given by such a tool. They no longer have to change data input or add new constraints and re-optimize if they feel a variable does not meet their value preference. They can just click and drag a flow / percent utilization variable and instantly see the changes to other variable values in real time as they drag. It allows them to explore much more rapidly the solutions space to find the one that best suit their preferences.

## *5.2. Customizable Generic Dashboard for Interactive Optimization*

Arabi et al. [27] presents a supply chain tactical planning problem of the forest products industry which has the form of what we showed in the previous section. For that problem, optimization was giving great results in comparison to the manual planning done by the company. Their managers were happy with the modeling of their supply chain but wanted the ability to personalize the solutions returned by the optimization software. Moreover, they required the ability to compare different solutions according to indicators specific to the company.

From there, we decided to explore an approach where any decision maker making use of a mathematical model could themselves build custom dashboards allowing for the evaluation and modification of solutions. We thus developed a generic Microsoft Excel plugin allowing for such a task. The plugin is encapsulating the multi-stage triangular approach and makes use of a standardize data format for the input part containing the results of the 2n required optimization (as explained in Section 3.1). Figure 18 shows the relationship between the components of that system. The current implementation allows optimization with either CPLEX or GLPK linear programming solver. The system is generic to any linear programming model, hence it takes as input either a model file (.mod) or the linear programming file in the standard LP format of CPLEX. The other input file is a text file for expressing the set of decision variables of interest for which we want to be able to change the solution value in real time. Preprocessing is done using these two files and the proper LP solver in order to generate the original solutions that are to be used later by the triangular method (box "Automatic solution adjustment" in Figure 18). These solutions are stored within a hidden tab in Excel. Another tab (the "Data" tab) shows, for each of the defined "variables of interest", their current value (from the first optimization setting the objective value), and their minimal and maximal possible value for the solution to stay in the optimality solutions space. When optimality tolerance are set, minimum and maximum values within these tolerances (ex: 95%; 90%; 85%) are also displayed.

From these data, the user can build charts and tables using standard Excel functionalities, linking to original cells from the "Data" tab. Then, when the decision maker is changing the value of a variable of interest in a cell linked to the original data, an automatic script ensures the value is still within the optimality tolerance. Then, an automatic solution adjustment is done by the plugin (using the triangular method with multi-stage approach) and all other variables of interest are updated in the "Data" tab, and therefore, so are the custom tables and charts linked to them.

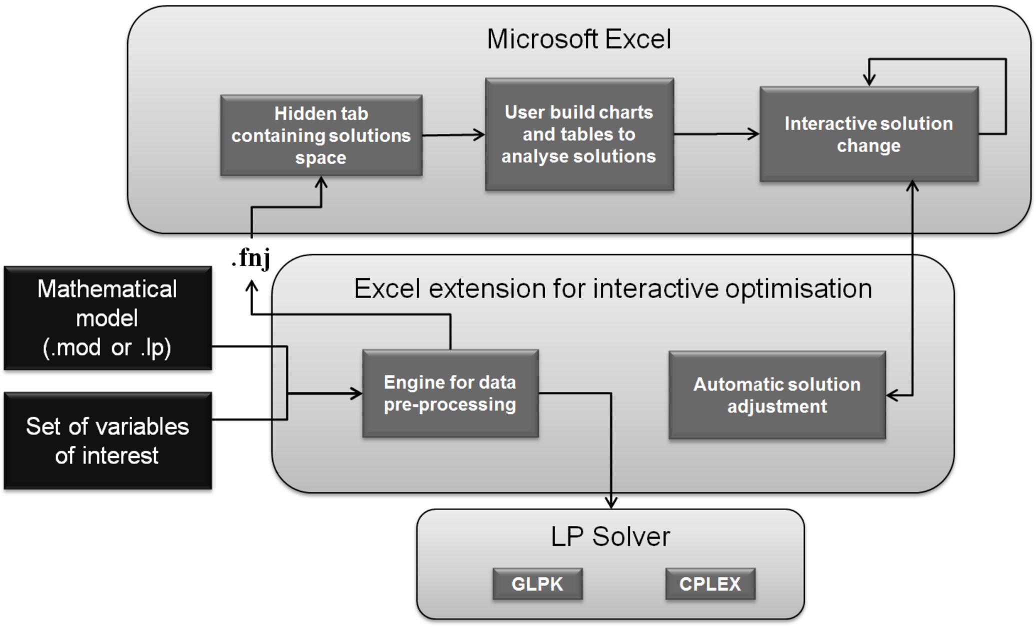

Microsoft Excel
Hidden tab containing solutions space
User build charts and tables to analyse solutions
Interactive solution change
.fnj
Mathematical model (.mod or .lp)
Excel extension for interactive optimisation
Engine for data pre-processing
Automatic solution adjustment
Set of variables of interest
LP Solver
GLPK
CPLEX

Figure 18. System for customizable interactive dashboard and decision support system.

## 5.3. *Applications*

We have used this technology in a many tactical planning projects where linear programming was involved to find optimal solutions.

It was first applied to the case described in Section 3.3.1. Figure 19 shows one of the sheet within the dashboard developed. It depicts the volume of lumber that should be moved between their mills and the volume that is to be sent to market from each location, per period. Numbers on the arcs are decision variables that could be changed manually by the decision maker if he decides to have more lumber coming from a mill instead of another one. Such a change triggers an immediate change in all of the other numbers. Traditionally, the decision maker would have to run the optimization again (which takes about 15 minutes in this particular case). With our Excel plugin, the change takes a fraction of a second to compute again a solution that is within optimality range and minimize the change of other variables.

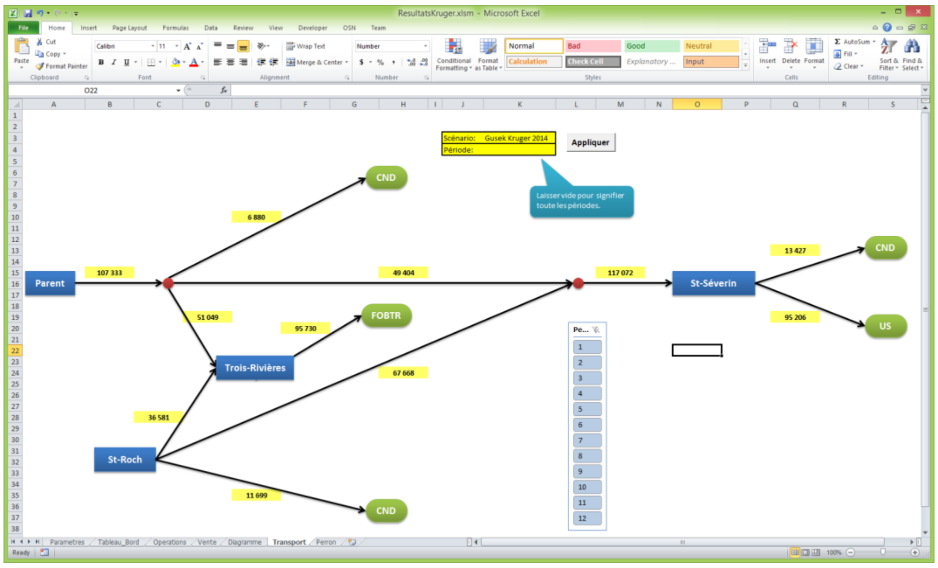
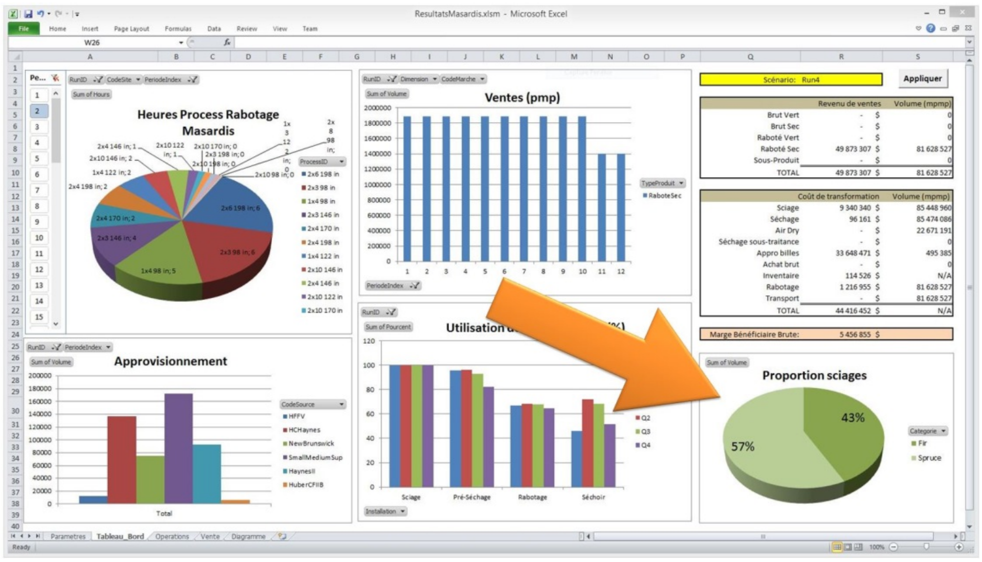


Figure 19. Dashboard of a lumber supply chain showing volume of lumber to be moved per period.

For the sawmill of another company, a key question was the proportion of spruce *vs* fur species to use to supply the mill. Spruce takes less time to dry and thus, its transformation to dry lumber is cheaper. On the other hand, it is more expansive to buy logs of spruce and it is not available in large quantity. Because of that, changing the spruce / fur ratio may change the solution (kiln dryer utilization, volume sold dry versus green, etc.) Allowing the decision makers to manually change the ratio and see the effect on the solution help them make a decision on how to supply the mill. Figure 20 shows the dashboard we made for the company. With the Excel plugin, they can change spruce / fur ratio and immediately see the impact on all other indicators.



Scénario: Run4

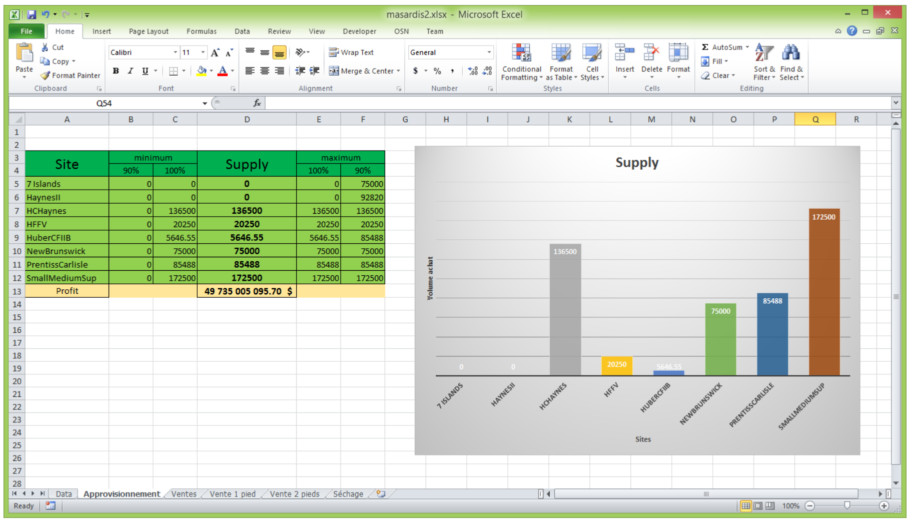
| | Revenu de ventes | Volume (mpmp) |
|---|---|---|
| Brut Vert | - $ | 0 |
| Brut Sec | - $ | 0 |
| Raboté Vert | - $ | 0 |
| Raboté Sec | 49 873 307 $ | 81 628 527 |
| Sous-Produit | - $ | 0 |
| TOTAL | 49 873 307 $ | 81 628 527 |

| | Coût de transformation | Volume (mpmp) |
|---|---|---|
| Sciage | 9 340 340 $ | 85 448 960 |
| Séchage | 96 161 $ | 85 474 086 |
| Air Dry | - $ | 22 671 191 |
| Séchage sous-traitance | - $ | 0 |
| Appro billes | 33 648 471 $ | 495 385 |
| Achat brut | - $ | 0 |
| Inventaire | 114 526 $ | N/A |
| Rabotage | 1 216 955 $ | 81 628 527 |
| Transport | - $ | 81 628 527 |
| TOTAL | 44 416 452 $ | N/A |

Marge Bénéficiaire Brute: 5 456 855 $

Figure 20. Dashboard of a lumber supply chain showing proportion of spruce / fur to replenish (bottom right pie chart), along with other indicators.

For the same mill, Figure 21 presents how log sourcing should be made according to one optimal solution. The table on the left shows the volume to supply from each site, along with minimum and maximum volume for the solution to stay within 90% of optimality. Current values are also displayed as a bar chart, on the right.



| Site | minimum | | Supply | maximum | |
|---|---|---|---|---|---|
| | 90% | 100% | | 100% | 90% |
| 7 Islands | 0 | 0 | **0** | 0 | 75000 |
| HaynesII | 0 | 0 | **0** | 0 | 92820 |
| HCHaynes | 0 | 136500 | **136500** | 136500 | 136500 |
| HFFV | 0 | 20250 | **20250** | 20250 | 20250 |
| HuberCFIIB | 0 | 5646.55 | **5646.55** | 5646.55 | 85488 |
| NewBrunswick | 0 | 75000 | **75000** | 75000 | 75000 |
| PrentissCarlisle | 0 | 85488 | **85488** | 85488 | 85488 |
| SmallMediumSup | 0 | 172500 | **172500** | 172500 | 172500 |
| Profit | | | **49 735 005 095.70 $** | | |

Figure 21. Dashboard to analyze change in sully sources.

More recently, we integrated the technology to an existing Excel spreadsheet used by a company to optimize harvesting plan and wood allocations to five of their mills (Figure 22). Each sector has to be harvested using one of the three possible harvesting modes (short stem, long stem, mixed). Two transport modes are possible. On the sheet showed in Figure 22, they can see in real time the impact of a manual change on the optimality of the solution.

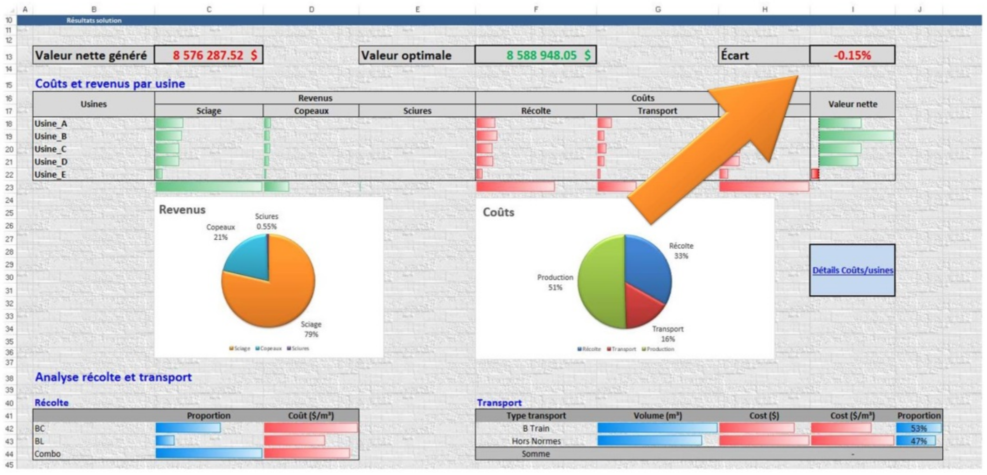


Figure 22. Dashboard to analyze cost and revenues of a solution.

We also applied this technology for a hardwood flooring production process described in Wery et al. [28]. The plant manager has to determine how much to supply from the different suppliers (given past production results using material from these suppliers), which recipe to run on the shop floor (using specific bundles from the suppliers) and how much of each product to produce. The objective is to maximize production value at minimum cost while satisfying constraints on products proportion to be able to satisfy customer demand with product variety in each box. This last planning problem is bidirectional as the decision maker may interactively change the volume supplied from the suppliers to see the impact on recipes to use and the different products made, or he can change the volume produced for different products and see the impact on the supply and recipe to run.

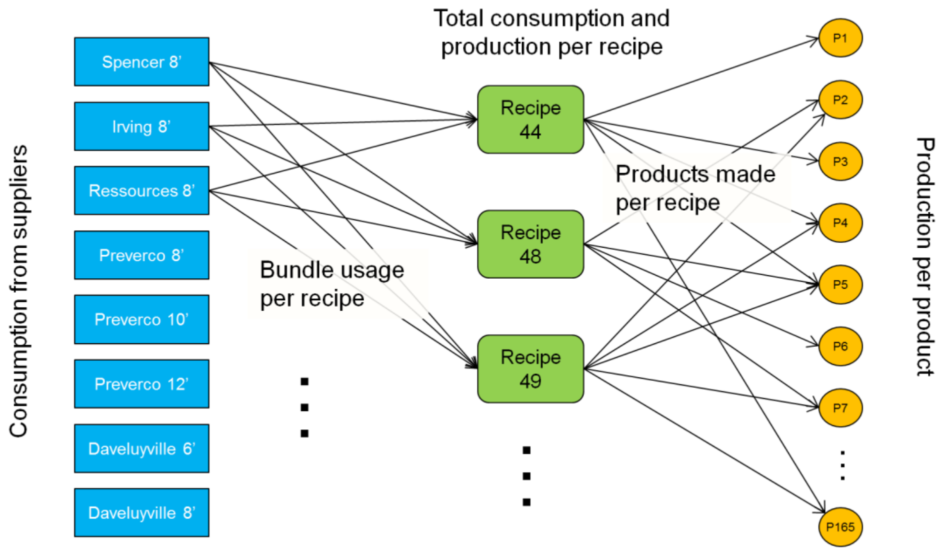


Figure 23. Decision variables for the optimization problem of a flooring factory.

## 6. Conclusion

Supply chain optimization leads to large problems with hundreds of thousands variables. They however can be solved in reasonable time (less than an hour) as they often show a linear structure. However, this computation time is unacceptable for iterative and interactive decision-making contexts where decision-makers need to provide their input.

We proposed a system that provides the decision-makers with a representation of an optimal solution space directly on the charts they are used to work with. This space is modeled as a convex hull of optimal solutions minimizing/maximizing the variables of interest. Users can interactively modify the value of a variable and the system is able to recompute a new optimal solution in less than a second (instead of optimizing the original problem, which is very long).

We developed four approaches for this real-time re-optimization (one exact approaches and three heuristics), among which the *triangular heuristic* performed very well, showing near-optimal performance. As many managers and decision makers are willing to sacrifice some percentage of optimality to increase the flexibility of a solution, we have introduced optimality tolerance and explored how it could be integrated into the triangular heuristic.

We proposed an interactive supply chain optimization interface upgrading LogiLab, a software developed for supply chain optimization in the forest-products industry. We also developed a customizable and generic dashboard system for interactive optimization and we presented the architecture for such systems. Through the diversity of the cases we made using these dashboards for interactive optimization, we were able to show that we can overcome the limitation of the single solution problem arising when using linear programming optimization. The plugin being compatible with Excel (which allows the decision makers to build its own dashboards), we believe it could be of a great benefit for the so many companies and consultants using linear optimization and Excel as a decision support tools.

As mentioned, the method used here only applies to linear problems. But many of the problems we face in the world do not follow that linearity. We are working at expanding the framework presented in this paper such as to treat mixed integer problems where the decision maker would change an integer decision variable.

**Acknowledgments**

This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC).

## References

[1] Hearst MA. Mixed-Initiative Interaction. IEEE Intelligent Systems. 1999; 14(5): p. 14.

[2] Allen JE. Mixed-initiative interaction. IEEE Intelligent Systems. 1999; 14(5): 14-16.

[3] Fleming M. The use of increasingly specific user models in the design of mixed-initiative systems. In: Proceedings of the 17th Conference of the Canadian Society for Computational Studies of Intelligence (LNCS #3060); 2004; London, Canada; p. 434-438.

[4] Klau G, Lesh N, Marks J, Mitzenmacher M. Human-guided search. Journal of Heuristics. 2010; vol. 16, p. 289-310.

[5] Benayoun R, de Montgolfier J, Tergny J, Laritchev O. Linear programming with multiple objective functions: Step method (stem). Mathematical Programming. 1971; 1(1): 366–375. DOI: http://dx.doi.org/10.1007/BF01584098

[6] Meignan D, Knust S, Frayret JM, Pesant G, Gaud N. A review and taxonomy of interactive optimization methods in operations research. ACM Transactions on Interactive Intelligent Systems. 2015; 5(3):17-43. DOI: http://dx.doi.org/10.1145/2808234

[7] Scott SD, Lesh N, Klau GW. Investigating Human-Computer Optimization. In: CHI 2002, April 20-25 2002, Minneapolis, Minnesota, USA.

[8] Yamada T, Sato T, Tomiyama T, Ueki, N. Evaluating Explanation Function in Railway Crew Rescheduling System by Think-Aloud Test. In: 5th IIAI International Congress on Advanced Applied Informatics, 2016. DOI 10.1109/IIAI-AAI.2016.93

[9] Pinedo ML. Design and implementation of scheduling systems: Basic concepts. In Michael L. Pinedo. Scheduling: Theory, Algorithms, and Systems. 4th ed. 2012. Springer. pp.459–483.

[10] Van Vliet A, Boender CGE, Rinnooy Kan AHG. Interactive optimization of bulk sugar deliveries. Interfaces. 1992; 22(3): 4–14. DOI:http://dx.doi.org/10.1287/inte.22.3.4

[11] Williams M, Gharbi H, Ulusan A, Ergun O, Xiaofeng Z, ZhangS, Harteveld C. Toward Human in the Loop Optimization Through Game-Based Experiments. In: CHI PLAY'16 Extended Abstracts, October 16-19, 2016, Austin, TX, USA.

[12] Kun W, Havens WS. Modelling an academic curriculum plan as a mixed-initiative constraint satisfaction problem. In: Proceedings of the 18th Conference of the Canadian Society for Computational Studies of Intelligence (LNCS #3501); 2005; Victoria, Canada; p. 79-90.

[13] Ai-Chang M, Bresina J, Charest L, Chase A, Hsu JCJ, Jonsson A, Kanefsky B, Morris PH, Kanna R, Yglesias J, Chafin BG, Dias WC, Maldague PF. MAPGEN: mixed-initiative planning and scheduling for the Mars Exploration Rover mission. IEEE Intelligent Systems. 2004; 19(1): 8-12.

[14] Bresina JL, Morris PH. Mission operations planning: beyond MAPGEN. In: Second IEEE International Conference on Space Mission Challenges for Information Technology; 2006; Pasadena, California; p. 477-484.

[15] Bresina JL, Morris PH. Mixed-initiative planning in space mission operations. AI Magazine. 2007; 28(2): 75-88.

[16] Guiost B, Debernard S, Poulain T, Millot P. Supporting air-traffic controllers by delegating tasks. In: Proceedings of the 2004 IEEE International Conference on Systems, Man and Cybernetics; 2004; The Hague , Netherlands; p. 164-169.

[17] Lenor T, Lewis M, Hahn S, Payne T, Sycarn K. Task characteristics and intelligent aiding. In: Proceedings of the 2000 IEEE International Conference on Systems, Man and Cybernetics; 2000; Piscataway, NJ, USA; p. 1123-1127.

[18] Linegang M, Haimson C, MacMillan J, Freeman J. Human control in mixed-initiative systems: lessons from the MICA-SHARC program. In: Proceedings of the 2003 IEEE International Conference on Systems, Man and Cybernetics; 2003; Washington, D.C.; p. 436-441.

[19] Smith SF, Lassila O, Becker M. Configurable, Mixed-Initiative Systems for planning and Scheduling. In: ARPI 1996 Proceedings. AAAI Press, Menlo Park, CA, 1996.

[20] de Kok AG, Graves SC. Supply Chain Management: Design, Coordination and Operation. Amsterdam: Elsevier, 2003.

[21] Marier P, Gaudreault J, Knaptik M, Savard M. Maximizing profits through sawmills specialization and supply chain design. In: Proceedings of the 14th Annual International Conference on Industrial Engineering Theory, Applications & Practice; 2009; Anaheim, CA; p. 484-491.

[22] Haartveit EY, Kozak RA, Maness TC. Supply chain management mapping for the forest products industry: Three cases from western Canada. Journal of Forest Products Business Research. 2004; vol.1; article no 5.

[23] Jerbi W, Gaudreault J, D'Amours S, Nourelfath M, Lemieux S, Marier P, Bouchard M. Optimization/simulation-based framework for the evaluation of supply chain management policies in the forest product industry. In: IEEE International Conference on Systems, Man, and Cybernetics (IEEE SMC 2012); 2012; Seoul, Korea; p. 1742-1748.

[24] Singer M, Donoso P. Internal supply chain management in the Chilean sawmill industry. International Journal of Operations & Production Management. 2007; 27(5): 524-541. DOI: http://dx.doi.org/10.1108/01443570710742393

[25] Dantzig, GB Linear Programming and Extensions. Princeton University Press, Princeton, New Jersey, 1963.

[26] Papadimitriou CH, Steiglitz K. Combinatorial Optimization: Algorithms and Complexity. Mineola, N.Y.: Dover, 1998.

[27] Arabi M, Gaudreault J, Nourelfath M, Favreau J, Morneau-Pereira M. Integrating Optimization and Simulation for Supply Chain Tactical Planning in the Forest Products Industry. In: International Conference on Information Systems, Logistics and Supply Chain (ILS); 2012 Aug 26-29; Québec, Canada.

[28] Wery J, Marier P, Gaudreault J, Chabot C, Thomas A. Improving A Hardwood Flooring Cutting System Through Simulation And Optimization. In: Proceedings of the 2015 Winter Simulation Conference; 2015 Dec 6-9; Huntington Beach, California.

**Vitae**

Jonathan Gaudreault is the co-director of FORAC, a research consortium dedicated to the development of decision support systems for the forest products industry. Over the last fifteen years he has led a number of key projects related to operations planning and scheduling, bringing together ideas from the artificial intelligence and operations research fields. He previously worked as an R&D manager in the software industry and was awarded as "One to watch" by the FIQ (OCTAS award). In 2012, Pr. Gaudreault and other researchers from the FORAC Research Consortium were recognized with the Brockhouse Canada Prize for Interdisciplinary Research in Science and Engineering, Canada's premier award for interdisciplinary research.

Claude-Guy Quimper obtained his Ph.D. degree in computer science at the University of Waterloo. He is an associate professor at Université Laval (Quebec City). His research interests lie in in the satisfaction and optimization of combinatorial problems using constraint programming. He designs filtering algorithms for global constraints that reduce the size of the search space and therefore the computation time required to find a solution. His work is applied to different fields such as planning, scheduling, rostering, and path finding.

Philippe Marier has been a research professional at FORAC since 2002. He is an Industrial Engineer with an MBA specialized in operations management and decision systems, Philippe has more than 15 years' experience in developing decision aid systems. He contributed to the design and development of many optimization algorithms for the forest products industry. His models are used in solutions for chip replenishment, sawmills strategic investments, transportation planning, sawmills shortterm planning and tactical supply chain design.

Mathieu Bouchard works at PetalMD where he leads the development of the mathematical engine of their medical scheduling software. He has also developed various optimization softwares for forest planning, supply chain management, transportation and scheduling. He worked for 3 years as research professionnal for the Forac Research Consortium.

François Chéné is a graduate student pursuing a master's degree in computer science with the FORAC research consortium under the Direction of Jonathan Gaudreault and Claude-Guy Quimper. He is also working part time as a videogame programmer since 2014.

Jean Bouchard is a master's degree student in computer science at Université Laval (Quebec, Canada). During his last baccalaureate year, he worked with the FORAC research consortium developping Excel tools for decision-makers. His master's project involves improving an Excel add-in for desision makers to explorer near optimal solutions space.