Les Cahiers du GERAD ISSN: 0711–2440

Shift scheduling under stochastic demand

R. Pacqueau
F. Soumis

G–2014–46

July 2014

Shift scheduling under stochastic demand

Rémi Pacqueau
François Soumis

GERAD & Département de mathématiques et de génie industriel, Polytechnique Montréal, Montréal (Québec) Canada H3C 3A7

remi.pacqueau@polymtl.ca
francois.soumis@gerad.ca

July 2014

Les Cahiers du GERAD
G–2014–46

Abstract: Shift scheduling when the demand for employees is stochastic is usually done in two steps. Since employees need to know their shifts before the demand is known (e.g. two weeks in advance), the shifts are scheduled using a forecast demand. Once the demand is known, a recourse—depending on the company and legislation—may be possible. For example, overtime can be added for some employees, part-time employees can be hired, or the breaks schedule can be changed. This is a deterministic approach. We propose an alternative method for scheduling work shifts, and we thoroughly study and quantify the savings. Stochastic programming is used for both steps (full-time shifts and recourse) using the stochastic distribution of the demand. The resulting problem is hard to solve: 10 million IP variables for a 96-period problem with 500 scenarios. We develop a heuristic that is fast (sometimes less than 15 minutes) and yields significant savings over deterministic solutions (from 0.5% to 15%, depending on the instance). The stochastic approach is worthwhile, since the computational time is low, the solution is nearly always better, and the deterministic approach turns out to be unstable. The stochastic approach presented here is worthwhile since the computational time is low and it provides better solutions in general (savings of up to 15%) than those of the deterministic approach, which can be unstable under stochastic demand.

Key Words: Stochastic programming, OR in manpower planning, Large scale optimization, L-Shaped method, Benders' decomposition.

Acknowledgments: Research supported by a NSERC Collaborative Research and Development grant no. RDCPJ411700-10 and Kronos Technologies.

1 Introduction

The shift scheduling problem when the demand for employees is deterministic has been well studied (see for example Baptiste *et al.* [1]). The problem has a time horizon discretised into a set of periods P with a demand of d_p employees for period $p \in P$. The objective is to cover the demand at minimal cost with a set J of feasible shifts. The set J and the cost c_j of shift $j \in J$ are defined by the work rules. Dantzig [2] uses a set-covering formulation with a variable per shift with a break schedule. Aykin [3] also uses a set-covering formulation, replacing the breaks in the shift definitions by break windows, thus using fewer shift variables. Other variables control the break allocation. Bechtold and Jacobs [4] use a similar formulation, but use *forward* and *backward* constraints to control the break allocation.

Generally, employees need to know their shifts several weeks in advance. In many contexts, the demand is not well-known at that time: for example, the demand in a supermarket or a retail store might depend on the traffic or the weather, and calls to a call-centre might vary depending on other random events. The classical way of dealing with this problem is to build the shifts in advance using the forecast demand. Once the demand is known (it is often known one day in advance in supermarket and retail stores, when a new forecast is prepared based on more accurate information), the company is allowed a *recourse* to fit the employee coverage to the actual demand: it can ask for overtime, hire part-timers, or slightly change the breaks (while staying within a specific time window). The recourse is usually more expensive per hour added than using full-time shifts; this is the case considered in this paper. This approach solves the *expected-value* (or *deterministic*) problem (EV). The recourse policy can be different in some sectors where the demand remains uncertain until the day of operation: it may involve assigning supervisors to work, asking workers to volunteer for time off, etc.

A first approach solves the *expected-value* problem (EV) which basically replaces the random elements with their expected values. The EV solution is always suboptimal (see for instance Birge and Louveaux [5]). For example, let us assume that adding personnel at the recourse time is extremely expensive, and that the average demand is close to the median left-skewed (the mass of the distribution is above the average value). Half the time, the scheduled full-time shifts will not cover The scheduled full-time shifts will then have a strong probability of not covering the actual demand, so the company must pay for an expensive recourse. In this case covering slightly more than the average demand with the full-time shifts leads to a solution that is better than the EV solution.

We will present the variants of the problem using the notation of Birge and Louveaux [5]. Instead of using an average demand, stochastic programming (also called the *here and now* (HN) approach) uses the full stochastic distribution of the demand. The objective is to minimize the cost of the full-time shifts built in advance and the expected cost of the associated recourse.

To briefly introduce stochastic programming, we use the notation of [5]. Let ξ be a random vector depending on a random event $\omega \in \Omega$. Let $x \in X$ be a primary decision variable, that is a variable associated with a decision that must be taken before the information is known, and $z(x, \xi(\omega))$ the cost of decision x when event ω occurs. The stochastic problem can then be written:

$$(HN)\ \min_{x \in X} \mathbb{E}_\omega(z(x, \xi(\omega))) \tag{1}$$

If $\bar{\xi}$ is the expected value of ξ, the expected-value problem is simply:

$$(EV)\ \min_{x \in X} z(x, \bar{\xi}) \tag{2}$$

Let us denote by $\bar{x}$ the optimal solution of (EV). Another important quantity is the *expected value of the expected-value problem* (EEV), given by:

$$EEV = \mathbb{E}_\omega(z(\bar{x}, \xi)) \tag{3}$$

This represents the total expected cost of the deterministic solution: the cost of the full-time shifts and the expected cost of the associated recourse. A basic but significant stochastic-programming result states that in the case of a finite probability space:

$$HN \leq EEV \tag{4}$$

This means that the stochastic problem always gives, *on average*, a better solution than the EV. The difference between the deterministic solution and the stochastic solution is called the *value of the stochastic solution* (VSS):

$$VSS = EEV - HN \tag{5}$$

VSS represents the expected savings achieved by using the HN approach instead of the deterministic approach. The stochastic problem for shift scheduling has been addressed by Bard et al. [6] in the context of the US Postal Service. However, they studied only a 3-state probability space, making the problem small enough to be solved by CPLEX, but not realistic for other applications. Moreover, the problem structure and constraints were specific to the USPS.

The aim of this paper is to present a method for solving large-scale stochastic shift scheduling problems, and to determine if it is worth the extra computational time. First, we adapt Aykin's formulation to a stochastic shift scheduling problem. To solve the problem, we discretise both time and the probability space. A day is divided into 96 periods of 15 minutes. The demand on each day is a random vector. Only a finite number of scenarios are considered, i.e. only a finite number of demands can occur each day. The number of scenarios considered is a key factor: the more scenarios, the better the approximation, but the larger the problem. Our simplifications enable the use of a classical stochastic programming method: the *L-shaped* method, developed by Van Slyke and Wets [7], based on Benders' decomposition [8]. Since the variables are integer in the recourse subproblems, we use a heuristic adaptation of the L-shaped method. It proves to be efficient in both computational time and solution precision. The solutions of several instances are compared to their respective "average" solutions (a more precise definition will be given). The results are then discussed, and several ways to improve the computational efficiency are presented.

This paper is organized as follows. In Section 2 we formulate the problem using Aykin's formulation. In Section 3 we solve the LP relaxation and describe our heuristic. In Section 4 we present computational results for several instances. In Section 5 we discuss the results and evaluate when our stochastic approach is worth the extra computational time.

2 Problem formulation

Since the breaks are part of the recourse, Aykin's set-covering formulation [3] is well suited to this problem. In this section, it is adapted to the stochastic approach and to the different kinds of recourse considered. The problem consists in building shifts for a 96-period day under stochastic demand.

The recourse decisions are:

- Hiring part-timers.
- Introducing overtime after full-time shifts.
- Allocating breaks. Each full-timer must have a break during the shift's break window. The breaks are allocated only in the second-stage problem; slight adjustments can then be made to adapt the employee coverage to the actual demand.The breaks are allocated only in the recourse problem; each full-timer must have a break during the shift's break window.
- Not covering a part of the demand (expensive decision).

We consider the mono-activity case because in our situation it is always possible to assign employees to activities when the total demand is satisfied, because there are enough trained employees for each activity. The multi-activity case is a more complex problem. It involves different recourses (assigning employees to activities on which they are less productive or more expensive).

The stochastic problem consists in computing the optimal full-time-shift allocation *and* the recourse for *every* demand scenario in a single problem, in order to minimize the total expected cost: that of the full-time shifts and the expected recourse (depending on the full-time shifts).

2.1 Problem instances

For the computational study, we use a set of instances from a retail store with differing demand distributions. The real costs are replaced by approximations in terms of the hourly cost of regular shifts. The real costs are confidential company information. Our approximations are precise enough to permit a good comparison of the deterministic and stochastic models, and the solution time is not sensitive to the exact values of the cost. We describe below the other characteristics of the shift scheduling:

Periods: We study a 24-hour day. It is split into 96 periods of 15 minutes each, which is the standard discretisation for such problems.

Scenarios: The number of demand scenarios depends on the instance. They are considered equiprobable, but scenarios with different probabilities will not affect the solution time. They are generated using Legrain's software [9]. The idea is to split the construction of the scenarios into two parts. The mean of the scenarios is obtained via the traditional deterministic approach. This permits us to use the expertise accumulated in the company to select the factors and weights for the forecast. The variability is given by a model developed by analysing the differences in each period between the forecast demand and the observed demand for a set of historical dates. Each scenario is the product of the forecast demand and a vector of multipliers, for each period. The vector of multipliers is the product of four types of perturbations: a uniform perturbation for the 96 periods, the same uniform perturbations for 16 consecutive periods, the same uniform perturbations for 4 consecutive periods, and perturbations for each period. The intensity of each perturbation follows a log normal distribution, and they appear on each possible day with a Bernoulli distribution. Reasonable parameters for the log normal and Bernoulli distribution were selected by analysing small sets of data. Larger sets of data will be necessary to obtain more accurate estimates of the parameters of these distributions.

Full-time shifts: Each shift is 8 hours. The break window is 90 minutes at the middle of the shift. There is a shift starting in every period that is at least 8 hours before the end of the day. There are 65 possible full-time shifts ($96 - 8 \times 4 + 1$). The hourly cost of one worker is 1 unit. The other costs are expressed using the same unit. The break is not paid (paying the break would just change the cost of full-time shifts).

Breaks: Each break lasts 30 minutes. Each full-time shift must have exactly one break, and this break must be within its time window. A break can begin in any period. There are 69 possible breaks.

Part-time shifts: Each part-time shift lasts 3 or 4 hours. Every possible part-time shift is considered; there are 166. The hourly cost is 1.25. No break is necessary.

Overtime: Adding 1 or 2 hours of overtime is possible *after* a full-time shift. The hourly cost is 1.5.

Figure 1 illustrates the problem.

2.2 Notation

The sets used in this problem are described below:

P:	set of periods
J:	set of full-time shifts
JP:	set of part-time shifts
H:	set of overtime shifts
K:	set of breaks
Ω:	finite set of random events (scenarios)

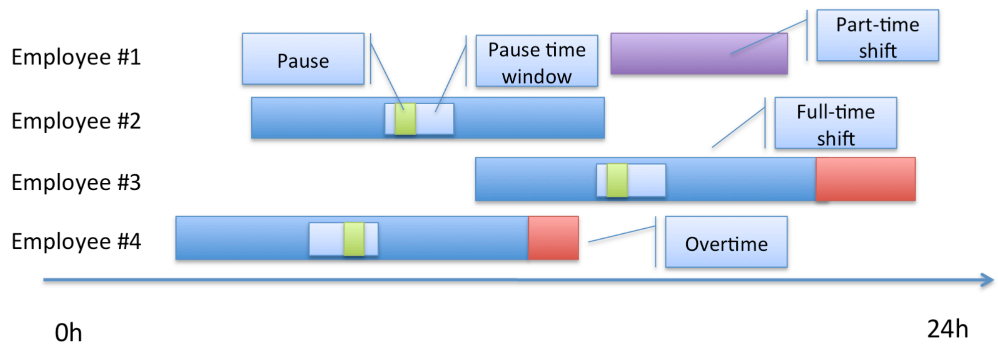


Figure 1: Graphical representation of the problem: Four employees assigned to four different kinds of shifts.

The problem parameters are the following:

c_j:	cost of full-time shift $j \in P$
cp_j:	cost of part-time shift $j \in JP$
cs_h:	cost of overtime shift $h \in H$
csc_p:	cost of not covering demand in period $p \in P$
p_ω:	probability of random event $\omega \in \Omega$
d_p^ω:	demand for employees in period $p \in P$ if random event $\omega \in \Omega$ occurs (integer parameter)
A_{jp}:	equals 1 if full-time shift $j \in J$ covers period $p \in P$; 0 otherwise
AP_{jp}:	equals 1 if part-time shift $j \in JP$ covers period $p \in P$; 0 otherwise
AK_{kp}:	equals 1 if break $k \in K$ covers period $p \in P$; 0 otherwise
AH_{hp}:	equals 1 if overtime shift $h \in H$ covers period $p \in P$; 0 otherwise
Q_{jk}:	equals 1 if break $k \in K$ is within break window of full-time shift $j \in J$; 0 otherwise
R_{jh}:	equals 1 if overtime shift $h \in H$ can be added to full-time shift $j \in J$; 0 otherwise

The decision variables are:

S_j:	number of full-timers assigned to full-time shift $j \in J$
SP_j^ω:	number of part-time employees assigned to part-time shift $j \in JP$ if event $\omega \in \Omega$ occurs
SH_h^ω:	number of employees assigned to overtime shift $h \in H$ if event $\omega \in \Omega$ occurs
B_k^ω:	number of full-timers assigned to break $k \in K$ if event $\omega \in \Omega$ occurs
X_{jk}^ω:	number of full-timers assigned to full-time shift $j \in J$ and assigned to break $k \in K$ if event $\omega \in \Omega$ occurs
XH_{jh}^ω:	number of full-timers assigned to full-time shift $j \in J$ and assigned to overtime shift $h \in H$ if event $\omega \in \Omega$ occurs
SC_p^ω:	demand not covered in period $p \in P$ if event $\omega \in \Omega$ occurs (no integrality requirement)

2.3 Formulation

The problem can then be formulated as follows:

$$\min \quad \sum_{j\in J} c_j S_j + \sum_{\omega\in\Omega} p_\omega \left[\sum_{j\in JP} cp_j SP_j^\omega + \sum_{h\in H} cs_h SH_h^\omega + \sum_{p\in P} csc_p \ SC_p^\omega \right] \tag{6}$$

$$\text{s.t.} \quad \sum_{j\in J} A_{jp} S_j + \sum_{j\in JP} AP_{jp} SP_j^\omega - \sum_{k\in K} AK_{kp} B_k^\omega \tag{7}$$
$$+ \sum_{h\in H} AH_{hp} SH_h^\omega + SC_p^\omega \geq d_p^\omega, \ \forall(\omega, p)$$

$$\sum_{k\in K} Q_{jk} X_{jk}^\omega - S_j = 0, \ \forall(\omega, j) \tag{8}$$

$$\sum_{j \in J} Q_{jk} X^{\omega}_{jk} - B^{\omega}_{k} = 0, \ \forall (\omega, k) \tag{9}$$

$$\sum_{h \in H} R_{jh} XH^{\omega}_{jh} \leq S_j, \ \forall (\omega, j) \tag{10}$$

$$\sum_{j \in J} R_{jh} XH^{\omega}_{jh} - SH^{\omega}_{h} = 0, \ \forall (\omega, h) \tag{11}$$

$$S_j, \ SP^{\omega}_{j}, \ SH^{\omega}_{h}, \ B^{\omega}_{k}, \ X^{\omega}_{jk}, XH^{\omega}_{jh} \in \mathbb{Z}^+, \tag{12}$$
$$SC^{\omega}_{p} \in \mathbb{R}^+, \ \text{for all } (j, \omega, h, k) \text{ only for relevant variables}$$

The objective (6) is split into a deterministic part, the cost of the full-time shifts, and an expected value, the expected cost of the recourse over all the scenarios $\omega \in \Omega$. Since Ω is a finite set, the expected value can be computed as a finite sum, leading to a classical (but huge) MIP problem. The full-time shift variables (S_j) are called *first-stage* variables. The recourse variables are called *second-stage* variables. The full-time shift variables (S_j) are called *first-stage* variables since these decisions are made without complete information on the random demand. The recourse variables on the other hand are called *second-stage* variables since these decisions are taken in response to the realisation of the random outcomes.

Constraint (7) ensures that the demand for employees minus shortages is met in every period of each scenario. Constraints (8)–(9) ensure that every full-time shift is allocated a break and that every allocated break is within the time window of its associated shift. Constraints (10)–(11) ensure this for the overtime shifts.

The number of full-time employees is not constrained because the same full-time shifts can be offered to volunteer part-time employees at the first stage.

The model does not contain break-scheduling variables for the planning stage. These are not necessary because there are no demand-covering constraints for this stage.

Observe that the model can be simplified if Equation (8) is substituted into Constraint (7) and into the objective function (6) to eliminate S_j. We did not do this because it increases the number of nonzero elements in the system: the variable X^{ω}_{jk} must be repeated for each period p in (6) and (7). We allow the presolve of CPLEX to decide whether or not to substitute. We observed few constraint eliminations. The same remark applies to constraints (9) and (11). The advantages of less dense constraint matrices for shift scheduling have been studied by Rekik et al. [10]; such matrices allow the solution time to be divided by two.

This formulation does not *seem* to be well-adapted to a Benders-like method, such as the L-shaped method, since there is no constraint involving only the primary variables (full-time shifts). Therefore, the first-stage problem will be totally free of constraints during the first iterations (except for nonnegativity and Benders' cuts). The master problem starts at the first iteration with a cost equal to zero, and 100% of the cost information must be transferred from the subproblem via Benders' cuts. We tried another formulation, adding extra constraints into the first-stage problem to start with a better lower bound on the cost, but computationally it was less efficient: it required fewer iterations but more time. We thus chose the first formulation. Finally, we will show that the Benders' decomposition works well even if 100% of the cost information must be transferred by Benders' cuts.

2.4 Problem size

Since a recourse must be computed for each scenario in the problem, the size is linear in the number of scenarios. The more scenarios, the more accurate the approximation of the stochastic distribution, and the more accurate the stochastic solution. The problem size is given in Table 1 for different numbers of scenarios.

Since every variable (except the under-covering variables) is integer, it is unlikely that commercial software such as CPLEX used as a standalone solver will be able to solve this problem in a reasonable time.

Table 1: Problem size.

# Scenarios	# Variables	# Constraints
25	462 012	12 800
100	1 847 865	51 200
200	3 695 995	102 400
500	9 239 065	256 000
1 000	18 478 065	512 000

3 LP and IP solution

3.1 LP solution

The aim of this subsection is to get a quick overview of the results in terms of savings and computational time that can be expected for the IP. The solution method used for the LP will also be the basis of the heuristic developed for the IP.

The method used for the LP is a well-known algorithm, called the *L-shaped* method, developed by Van Slyke and Wets [7]. This is an adaptation of Benders' decomposition [8] with a recourse function for each scenario. The idea is to decompose the problem into a first-stage problem containing the first-stage variables and a secondary problem containing the second-stage variables. The master problem involves the s_j variables and can be formulated as follows:

$$\min \sum_{j\in J} c_j S_j + \sum_{w\in\Omega} p_w Q^w(S) \tag{13}$$

where $Q_w(S)$ is the cost of the optimal recourse decision if the first-stage variables have value S and event w occurs (with probability p_w). The second-stage problem involves the variables $SP_j^w, SH_h^w, B_k^w, X_{jk}^w, XH_{jh}^w$, and SC_p^w. It splits by scenario and minimises the recourse cost for each scenario w (the [ ] part of (6)) under the constraints of this scenario (the constraints with index w in (7) to (11)). The integrality constraints in (12) are replaced by positivity constraints; the integrality requirement will be discussed later. When there is no integrality requirement on the second-stage variables, every function $Q^w(S)$ is a convex, piecewise linear function of S variables. These functions can be expressed in problem (13) as a set of inequality constraints, to define problem (14)–(16):

$$\min \quad \sum_{j\in J} c_j S_j + \sum_{w\in\Omega} p_w \theta_w \tag{14}$$

$$\theta_w \geq f_k^w(S)\ \forall\, k = 1, ..., K^w, \forall w \in \Omega \tag{15}$$

$$\theta \geq 0, \theta \in R. \tag{16}$$

The θ_w variables are the Benders' approximation variables associated with scenario $w \in \Omega$. The principle of the L-shaped method is to start without constraints (15) and to iteratively add these inequality constraints (also called Benders' optimality cuts.) At each iteration, a first-stage solution is found, then the second-stage problem associated with this first-stage solution is solved. If for scenario w the cost in the subproblem is greater than the cost in the master problem, we add a cut for this scenario. The coefficients of the variables S in the added $f_k^w(S)$ are given by the dual variables of the subproblem. Let $\alpha_p^w, p \in P$, $\beta_j^w, j \in J$, and $\gamma_j^w, j \in J$ be the dual optimal variables associated with constraints (7), (8), and (10) in the subproblem of scenario w. The cut is:

$$\theta^w + \sum_{p\in P}\sum_{j\in J} \alpha_p^y A_{jp} S_j - \sum_{j\in J}(\beta_j^w + \gamma_j^w) S_j \geq \sum_{p\in P} \alpha_p^w d_p^w. \tag{17}$$

Even if the number of Benders' cuts needed to exactly define the Q^w function is extremely large (one for each extremal point of the subproblem), only a few of these cuts are needed to compute the optimum, since not all the regions of the feasible space need to be precisely described. This multicut version of the L-shaped method has been used: at each Benders' iteration, one optimality cut is added for each scenario if necessary. With the multicut version more cuts are added and the master problem grows more rapidly. However, fewer iterations are necessary. The final problem is slightly larger but the total solution time is much smaller. The improvement in the number of iterations dominates. Figure 2 compares standalone CPLEX and the multicut L-shaped algorithm for the problem studied.

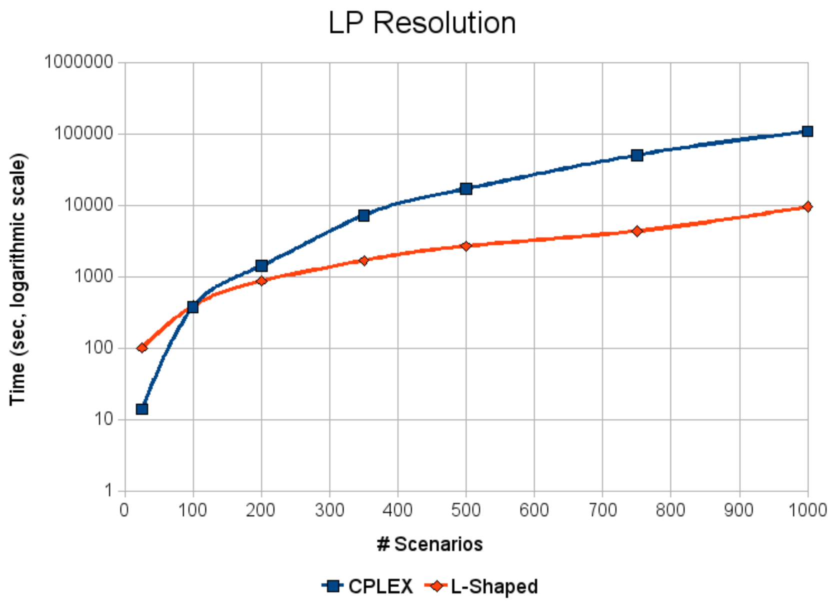


Figure 2: Solution time for LP relaxation: Comparison of standalone CPLEX 12 (with presolve) and multicut L-shaped method.

The computations are performed using servers with AMD Opteron 275 (2.2 GHz) processors and 8 GB of RAM. The programs are implemented in Java. CPLEX 12 is used either as a standalone solver (all default options activated, including presolve) or as a tool to solve the first-stage problem and second-stage L-shaped-method problems.

With only a few scenarios, CPLEX is ten times faster than the L-shaped method, but the trend is reversed when the number of scenarios is high, where the L-shaped method appears to be ten times faster than standalone CPLEX. For example, for 1000 scenarios, the LP relaxation is solved in 30 h with CPLEX and 2 h 40 with the L-shaped method.

3.2 Determining the number of scenarios

Since the number of scenarios is a key factor, it is important to choose it carefully, keeping in mind that it should be as large as possible.

To choose the number of scenarios, we use Legrain's work [9] to generate 10 000 scenarios based on the same forecast (or average) demand. This demand generally varies between 0 and 40 employees. The 10 000 scenarios are considered to represent the entire stochastic distribution. To check the impact of the number of scenarios on the solution process, the following procedure is applied 50 times for different values of $N << 10\,000$:

1. Choose randomly N scenarios among the 10 000 possible.
2. Solve the stochastic problem considering only those N scenarios.
3. Evaluate the "total cost" of the solution, that is the sum of the cost of the full-time shifts and the expected cost of the associated recourse over the *entire* 10 000 scenarios. (This implies computing the recourse for each of the 10 000 scenarios, which are simple and independent problems.)

Once the fifty total costs are known, the mean and standard deviation of this series is computed. A large standard deviation would mean that the choice of the scenario has a large impact on the result of the stochastic problem; this must be avoided. We also compute the cost of the *deterministic* and *wait-and-see* solutions. The *wait-and-see* approach involves waiting for the actual demand before setting the first- and second-stage variables. This approach is of course always better than the deterministic or stochastic approaches.

Figure 3 shows the results. As expected, the more scenarios, the better the total cost and also the standard deviation. Twenty-five scenarios appears to be a good choice for this instance, but standard deviations of up to 7% appeared for other instances. Thus, the use of more scenarios, and hence the L-shaped method, is inevitable. We set the number of scenarios to 500: the computational time is reasonable (around 1 h for the LP) and the solution does not have a strong dependence on the scenarios chosen.

3.3 Heuristic description

Some exact methods have been developed to solve stochastic programs with integrality requirements on the second-stage variables; see for example Sherali and Zhu [11] and Hamdouni et al. [12]. However, none of them is well-suited for this stochastic shift scheduling problem, because they are inefficient on large problems or apply only to specific problems.

To understand our heuristic, one has to keep in mind that the L-shaped method iteratively constructs an approximation of the recourse function, which is piecewise linear. Indeed, the whole set of optimality cuts represents the entire recourse function. However, only a few of the cuts must be generated in the L-shaped resolution, hence the speedup. Figure 4 illustrates this link between the optimality cuts and the recourse function.

The main idea of our heuristic is to approximate the recourse function[1] by a generated subset of Benders' optimality cuts, and to use this approximation in a classical branch-and-bound scheme to get the first-stage IP variables. The heuristic is shown in Figure 5.

Our heuristic scheme first solves the LP relaxation of the problem (boxes 1 to 4) to get a good approximation of the recourse function using the generated Benders' cuts.

To get even more cuts, all the first-stage variables with a fractional part greater than 0.8 are rounded up (boxes 5 to 6). The LP is then solved again with these variables fixed. This is fast since we keep the previous Benders' cuts: usually fewer than ten L-shaped iterations are needed to compute a new optimal solution. This fixing and solving process is iterated until there are no more variables to fix (generally five iterations suffice). The goal of fixing the variables which are likely to be at the upper integer in the optimal IP solution is to generate further Benders' cuts in order to get a better approximation of the recourse function.

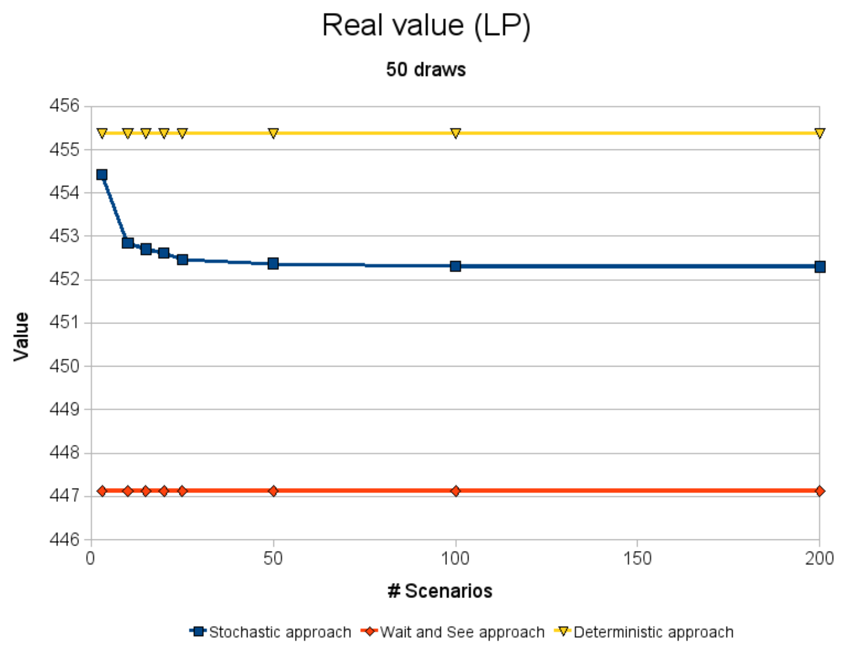
Some first-stage variables remain unfixed. The first-stage problem (see Birge and Louveaux [5] for the terminology) containing the first-stage variables and all the accumulated Benders' cuts is as follows:

$$\begin{aligned} \min \quad & \sum_{j \in J} c_j S_j + \sum_{\omega \in \Omega} p_\omega \theta_\omega && (18) \\ s.t. \quad & \text{Fixed variables} && (19) \\ & \text{Benders' cuts} && (20) \end{aligned}$$

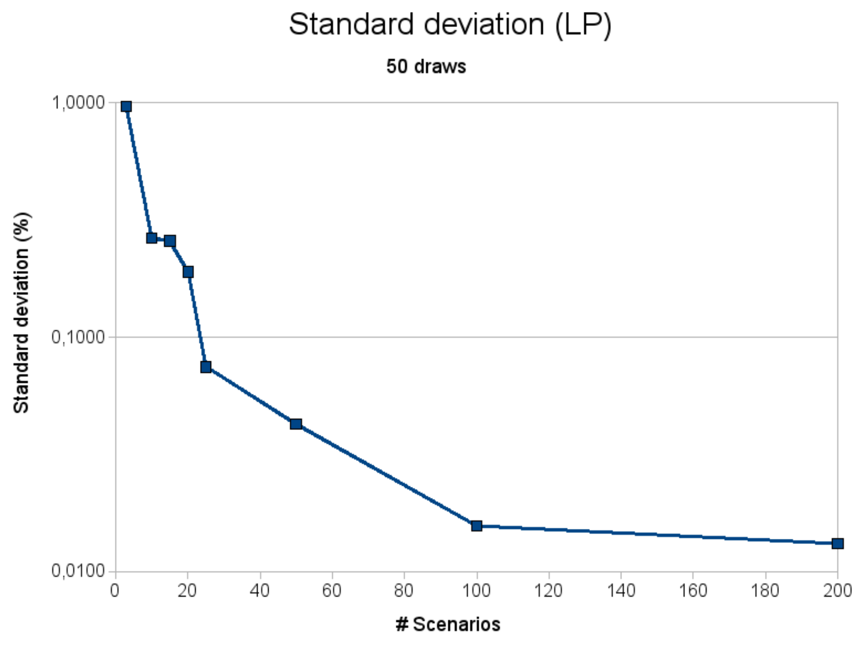
This problem is relatively small (65 shift variables and 500 recourse variables), and can be easily solved via CPLEX using a small MIP tolerance (0.1%) (box 7). Solving this problem corresponds to approximating the recourse function by its minoring Benders' cuts. The more Benders' cuts, the better the approximation.

Since the Benders' cuts are lower bounds on the recourse function, the objective function will be smaller than the objective value of the real problem. This approach does not explicitly compute the second-stage variables. In practice they can be easily calculated once the demand is known (one scenario to evaluate). In the research phase, after the algorithm, we solve to integrality the subproblems for every scenario to evaluate the underestimation of the recourse given by the algorithm.

[1] That is, the expected cost of the recourse.



(a) Total cost



(b) Standard deviation (logarithmic scale)

Figure 3: Total costs (mean) and standard deviations of solution using here-and-now (stochastic) approach for 50 draws of N scenarios from the 10 000, used to help choose number of scenarios.

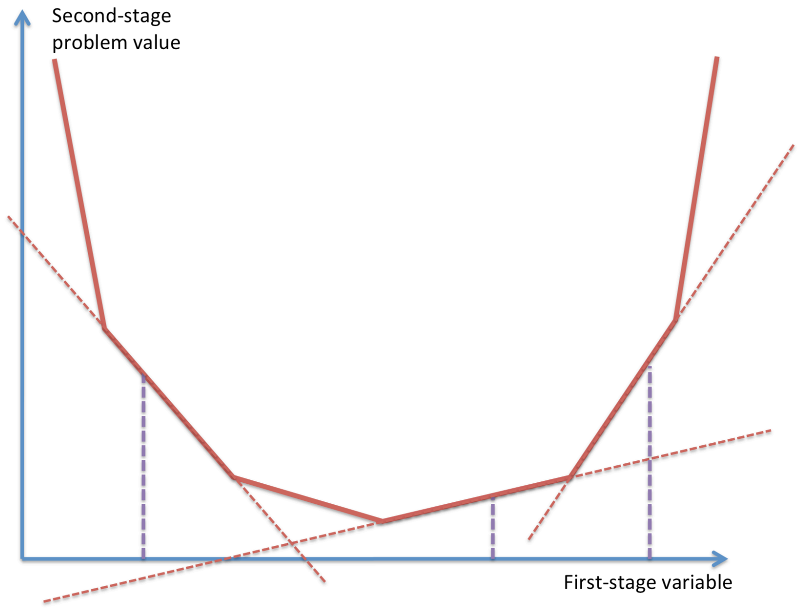


Figure 4: L-shaped method. The recourse function is the red line. At each iteration, a value of the first-stage variable is found. It yields a linear cut (dashed red line) corresponding to part of the piecewise-linear recourse function.

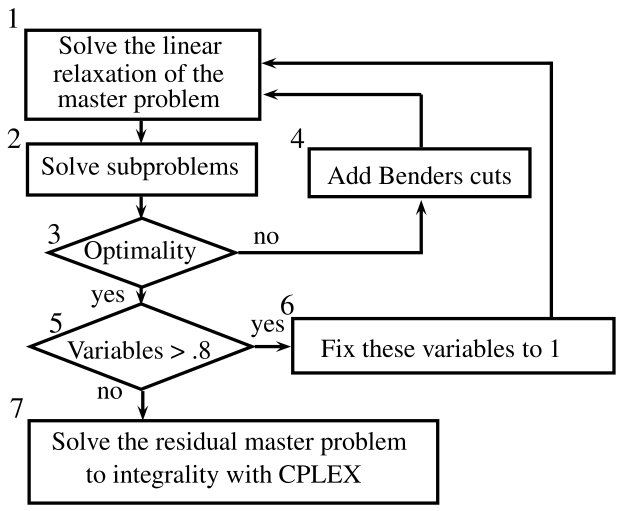

1
Solve the linear relaxation of the master problem
2
Solve subproblems
4
Add Benders cuts
3
Optimality
no
yes
5
Variables > .8
yes
6
Fix these variables to 1
no
7
Solve the residual master problem to integrality with CPLEX

Figure 5: Heuristic description. Benders' cuts are iteratively generated by solving the LP relaxation and some first-stage variables are fixed. The process concludes with a classical branch-and-bound scheme.

Two approximations are made in this heuristic: the recourse is always considered to be LP instead of IP, and it is approximated using Benders' cuts. However, the recourse cost is often small compared to the cost of the full-time shifts. Moreover, the integrality gap was small for the instances tried (1% or less), so considering LP variables for the second-stage problem is not an important issue.

4 Results

4.1 Toy model

To compare the heuristic to the optimal IP stochastic solution, we use a smaller model, with 100 000 to 800 000 variables depending on the number of scenarios (100 to 500). The flexibility in the shift definitions was reduced compared to the original problem presented in Table 1. This reduces the sets J, JP, H, and K. The numbers of the variables S, SP, SH, and B are reduced linearly. The numbers of the variables X and XH are reduced quadratically. We find the optimal and heuristic solutions and then compute their distances (see Definition 1) and their total costs (over 10 000 scenarios, as in Section 3). We also compute the deterministic solution. The results are presented in Table 2.

Definition 1 *The distance between two full-time-shift allocations S^0 and S^1 is defined by*

$$\sum_{j\in J} |S_j^0 - S_j^1|.$$

Table 2: Toy model: Comparison of exact and heuristic solutions. Exact solutions calculated by standalone CPLEX without any MIP tolerance, total cost evaluation with a 0.5% tolerance for each scenario. D1 is the distance between the optimal IP solution and the heuristic IP solution. D2 is the distance between the optimal IP solution and the optimal *deterministic* IP solution.

# Scenarios	Exact cost	Total cost	Gap (%)	D1	D2	VSS (%)
100	581.986	581.986	0	0	26	5.8
200	581.686	581.680	-0.001	2	26	5.8
500	581.679	581.725	0.01	1	26	5.8

The results show that the maximum distance found with our toy model is 2, in the 200-scenario instance. In this case the total cost provided by the heuristic is slightly better than that of the optimal stochastic solution. In the 100-scenario instance, the heuristic solution is the optimal solution. Moreover, the maximum optimality loss (0.01%) is negligible in comparison with the VSS (5.8%). Therefore, this heuristic appears to be accurate, and can be tested on large instances that CPLEX cannot solve alone.

4.2 Larger instances

Although there is no explicit link between the variance of a distribution and the associated VSS, we found empirically for our problem that the VSS was likely to be high if the variance of the data-set was large. Therefore, the results will be sorted according to the average variation coefficient of their respective datasets.

Definition 2 *The* average variation coefficient Γ *of a scenario set Ω is defined by*

$$\Gamma = \frac{1}{|P|}\left[\sum_{p\in P}\left(\frac{1}{\bar{d}_p}\frac{1}{|\Omega|}\sqrt{\sum_{\omega\in\Omega}(d_p^\omega - \bar{d}_p)^2}\right)\right]$$

where $\bar{d}_p$ is the average demand at period $p \in P$. This represents the average, over all periods, of the standard deviation divided by the average demand.

We give in Table 3 a brief description of the ten instances tested. The first three sets are real data from large retail stores. M4 to M10 are variants obtained by increasing or decreasing the variation in the scenarios or changing the duration of the perturbations. For instances M7 to M10, less noise means fewer short (15 minute) perturbations, and more long perturbations, to maintain approximately the same average variation coefficient.

To conclude this subsection, Table 4 lists the tolerances used for the CPLEX branch-and-bound scheme. Since the recourse generally costs around 10% to 20% of the overall cost, a branching precision of 0.5% in the recourse will perturb the overall cost by 0.05% to 0.1%.

Table 3: Description of ten instances used in experiments, all generated using Legrain's work [9].

Name	Γ (%)	Description
Standard	6.1	Instance used in Section 3.
Double peak	68	Instance with double-peak forecast demand.
M3	68	Instance with sharp forecast demand.
M4	34	Same as standard but larger average variation coefficient.
M5	45	Same as M3, Γ smaller.
M6	33	Same as double peak, Γ smaller.
M7	42	Standard forecast demand, less noise in random data.
M8	49	Less noise.
M9	32	Less noise.
M10	57	Less noise.

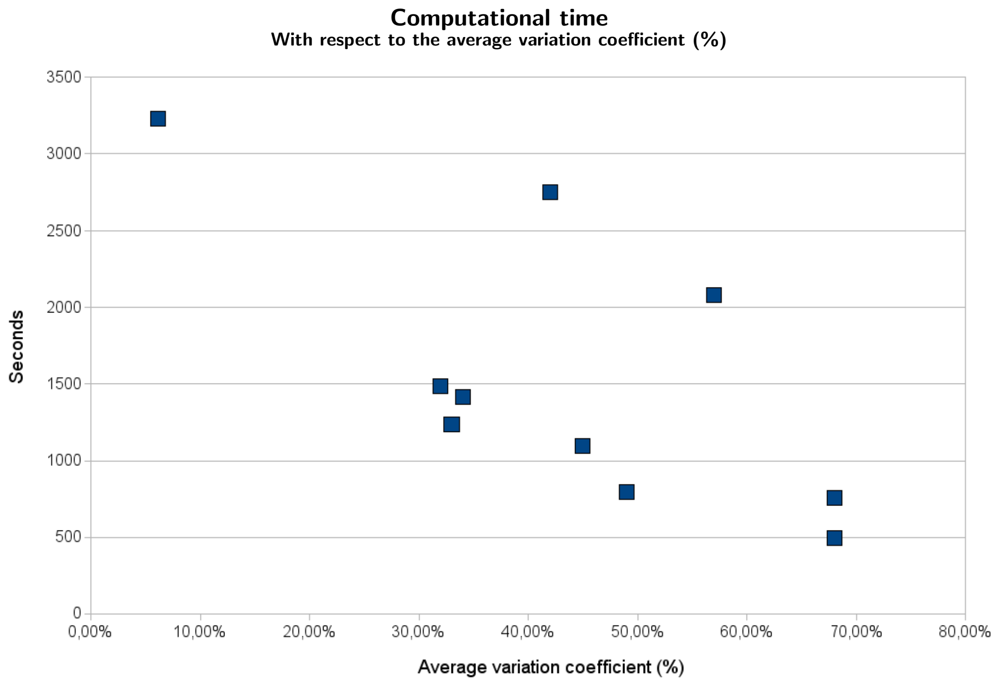
Table 4: Tolerance used in experiments.

Tolerance	Value (%)
BB final-phase heuristic	0.1
BB total-cost evaluation IP	0.5
BB deterministic solution IP	0.2

4.3 Computational time

Figure 6 shows the computational time on the AMD Opteron 275 servers.

Figure 6: Heuristic computational time for average variation coefficient for ten instances.



The computational time decreases as the average variation coefficient increases. It is almost always below 2000 s, and below 1000 s in favourable cases. The long solution times are caused by extra Benders' iterations. These have a dramatic effect on the computational time since the more iterations, the more Benders' cuts and the larger the first-stage LP problem that CPLEX must solve for each new iteration.

4.4 Deterministic solutions and test protocol

In the previous sections, we have referred to *the* deterministic solution and *the* value of the stochastic solution, to simplify the discussion. However, there are often several deterministic solutions with the same cost for the full-time shifts. Since these solutions are different, their expected total costs, i.e. the costs of the full-time shifts plus the expected cost of recourse, will differ. The difference between the deterministic and stochastic approaches is therefore not well defined. To illustrate this phenomenon, Figure 7 shows the relative variability of the total cost of the deterministic solutions.

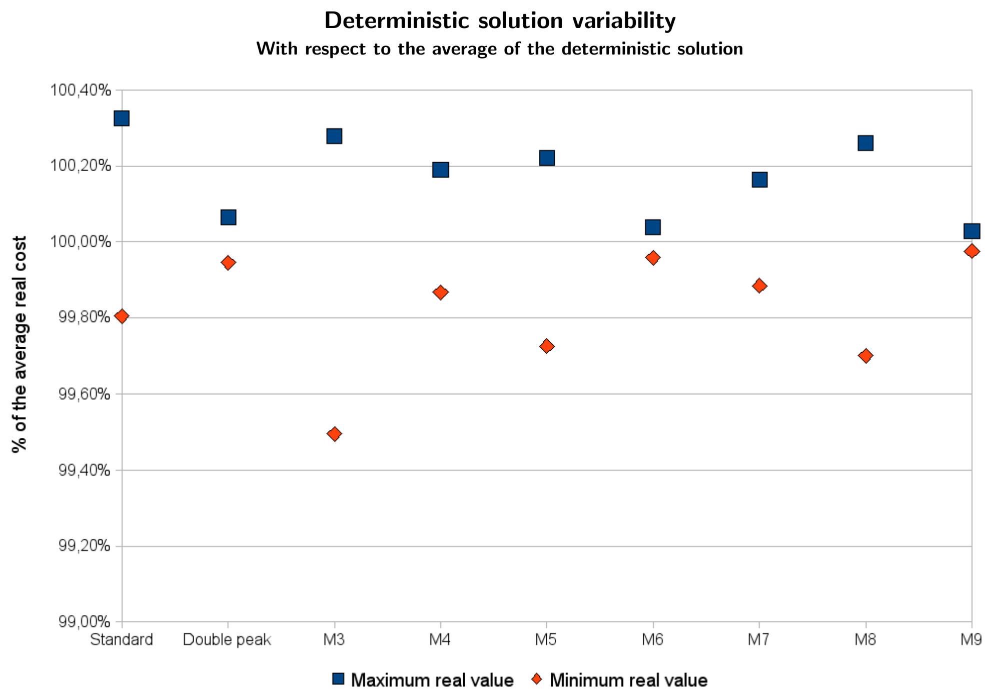


Figure 7: Variability of deterministic solutions. The maximum and minimum total costs are plotted as a percentage of the average deterministic total cost.

To get these maximum and minimum total costs, the following procedure was applied. For a given instance, a random perturbation of the full-time-shift costs is applied to the problem (about $10^{-5}\%$). This has a negligible impact on the cost of the full-time shifts, but it differentiates the optimal solutions. The total cost (i.e. the cost of the full-time shifts plus the expected cost of recourse over 10 000 scenarios) is then computed. This operation is repeated twenty times. The maximum and minimum total-cost values (relative to the average total costs of the deterministic solutions) are then plotted. The results confirm *a posteriori* that the $10^{-5}\%$ cost perturbations have a negligible effect on the cost of the full-time shifts in relation to the total cost variation.

This variation has the same order of magnitude as our branch-and-bound tolerance. It is not clear how much is due to variability and how much to branching errors. However, given these considerations, from now on the total cost and the VSS will be calculated as an average of twenty perturbed deterministic solutions.

4.5 Solution quality

To check the validity of the stochastic approach, we carry out the following procedure for every test instance and present the results in Table 5:

1. Compute the heuristic solution over the first 500 scenarios of the data set. The "computed cost" of this solution appears in line 1 of Table 5.

2. Compute a better evaluation of the cost of this heuristic solution. This expected recourse cost is evaluated by solving the LP-relaxation of the 500 recourse subproblems. This value is greater than the computed cost because the heuristic uses a lower bound on the recourse function. The "recourse underestimation (%)" is presented in line 2 of Table 5.
3. Compute a more exact total cost of the heuristic solution. The expected recourse cost is evaluated by solving to integrality the subproblems of 10 000 senarios. This "more exact" cost is presented in line 3 of Table 5.
4. Compute the expected cost of the deterministic solution by solving to integrality the recourse subproblems of 10 000 scenarios.
5. Find the "VSS heuristic" difference of the expected cost of the deterministic solution (computed in 4) and the expected cost of the heuristic stochastic solution (computed in 3). The VSS heuristic (%) presented in line 4 of Table 5 is a percentage of the cost of the stochastic solution.
6. Find the cost of the optimal stochastic programming solution when integrality is relaxed in the master problem and the subproblems. We relax integrality because it is impossible to solve to optimality, in a reasonable time, the stochastic programming problem with the integrality requirement. Find the cost of the deterministic solution when integrality is relaxed in the master problem and the subproblems. Compute the "VSS optimal LP" difference between these costs. This is a good estimate of the VSS with an optimal IP solution if the integrality gaps are small or are similar in the stochastic and deterministic problems. The "captured saving (%)" presented in line 5 of Table 5 is the percentage of VSS heuristic vs VSS optimal LP.

Table 5: Results on the quality of the solutions of the heuristic algorithm.

Instance	Std.	D. Peak	M3	M4	M5
Heuristic solution: computed cost	451.276	577.234	591.110	480.303	479.379
Recourse underestimation (%)	0.12	0.09	0.08	0.10	0.08
Heuristic solution: more exact cost	456.509	570.660	578.705	472.796	490.002
VSS heuristic (%)	0.18	6.81	8.38	0.28	0.12
Captured savings (%)	33.93	98.09	96.43	55.99	63.43

Instance	M6	M7	M8	M9	M10
Heuristic solution: computed cost	481.805	473.584	526.905	484.600	511.017
Recourse underestimation (%)	0.18	0.10	0.12	2.10^{-3}	0.10
Heuristic solution: more exact cost	482.568	475.927	513.258	489.975	516.398
VSS heuristic (%)	0.14	0.30	2.72	0.59	15.32
Captured savings (%)	27.24	51.00	85.15	98.01	98.01

The results in Table 5 are discussed in more detail in Section 5.2.

5 Discussion

5.1 Computational time

For most of the instances, the computational time is less than 15 min. For the worst case, this time is still less than an hour. It must be kept in mind that the original problem had 10 million IP variables, so 1 h is an acceptable computational time.

For difficult instances, the computational time is equally split between the second-stage problems and the first-stage problem. More precisely, the first-stage problem can initially be solved quickly, but its size increases at each iteration, so it is solved slowly after the first 100 iterations. However, the computational times for the second-stage problems are similar throughout the computation. For easy instances, only a few Benders' iterations are needed (just 23 for M3). The first-stage problem is small even at the end of the process: most of the (low) computational time is used to solve the second-stage problems.

The heuristic is therefore sufficiently fast.

5.2 Solution quality

There are two trends in the solution quality. For some instances, the savings are important: from about 3% to more than 15%. These instances are generally the fastest to solve. Indeed, for these instances, the recourse cost is quite important compared to the overall cost: the generated Benders' cuts are more selective than are those where the recourse is less important. The L-shaped method is therefore more efficient. In these instances, the captured savings approach 100%: the IP savings are similar to the LP savings, showing that the heuristic is efficient.[2]

For the other instances, the method does not yield significant savings. However, although the relative savings are small (below 1%), using the stochastic approach prevents the multiple optimal solutions that can occur in the deterministic case. Multiple deterministic solutions lead to uncertainty about the total cost of the deterministic approach: this cost will vary depending on how the optimal solution is computed. A 0.5% maximum gap has been measured on our instances (see Figure 7): This is an important source of error. The proposed method therefore provides a solution that is almost always better than the deterministic solution, and that is *stable*.

The stochastic approach is thus of interest. For easy instances, the computational time is generally below 15 min and the savings are worthwhile (up to 15%), and for harder instances, although the average savings are small, the computed solutions are *stable*: there is no variability resulting from multiple solutions. The computational time for such instances is acceptable.

6 Conclusion

6.1 Contributions

The contributions of this paper are as follows:

1. Shift scheduling has been carried out using an approach new to this field (stochastic optimization).
2. A fast and accurate heuristic has been developed to solve the problem.
3. The value of this stochastic approach has been quantified by a series of measurements.

Stochastic approach Most procedures for shift scheduling under uncertainty use a deterministic approach, because this is a simple and fast way to get a reasonable solution. The stochastic approach has not yet been used (except by Bard *et al.* [6], but for only three scenarios and in a context that is specific to the US Postal Service), because of the size of the problems. However, even if the stochastic problem we solved originally had 10 million IP variables, the results show that this approach is worth the extra computational time.

Heuristic The heuristic developed in this paper is both fast and accurate. CPLEX needs 7 000 s to solve a stochastic IP problem (instance "std") with only 25 scenarios and a 0.2% MIP tolerance. Our heuristic takes about 3 200 s for 500 scenarios. It gives a better solution than that based on 25 scenarios, and the choice of the scenarios is less important. Moreover, Section 4.1 shows that the heuristic is accurate. Table 5 indicates that the underestimation of the recourse by the heuristic is always less than 0.2%: this gives a good approximation of the loss of optimality due to the heuristic.

Value of this approach Instead of focusing only on the solution process, we quantify the savings introduced by the stochastic method, using the total-cost approach. When the average variation coefficient is small, the *value of the stochastic solution* is small: on average the difference between the stochastic and deterministic solutions is small. However, since the deterministic problem yields multiple optimal solutions, giving total costs that differ by about 0.5%, it should be avoided. The stochastic solution is unique and thus introduces

[2]However, the maximal IP savings are not necessarily smaller than the LP savings.

stability, for a reasonable computational time. When the average variation coefficient is large, and/or there is not too much noise in the scenarios, the value of the stochastic solution is up to 15%. Moreover, the computational times in these cases are relatively small (below 15 min). The stochastic approach, combined with the heuristic developed, is definitely valuable for shift scheduling under uncertain demand.

6.2 Possible improvements

Speedup The heuristic could be speeded up in two ways.

First-stage problem. If there are too many Benders' iterations, the first-stage problem becomes large and this dramatically slows down the solution process. We may be able to remove cuts during the process to keep the first-stage problem small. Since the Benders' cuts are approximations of the recourse functions, it is likely that some of the cuts are associated with useless regions of the function; they could be removed without affecting the solution process. If useful cuts are deleted, they will be generated again in later iterations.

Second-stage problems. Since the second-stage problems are independent, they can be solved on different processors without affecting the overall solution process. This would give significant speedups.

More steps in the decision process Another idea would involve using more decision stages: a first stage for full-time shifts with an uncertain demand, a second stage for part-time shifts with a more precise but still uncertain demand, and a final stage for the other recourses using the actual demand. This would of course considerably increase the problem size.

References

[1] P. Baptiste, V. Giard, A. Haït, F. Soumis, Gestion de production et ressources humaines, Presses internationales Polytechnique, 2005, Ch. 4: Gestion des horaires et affectation de personnel.

[2] G. Dantzig, A comment on Edie's "Traffic delays at toll booths", Journal of the Operations Research Society of America 2 (3) (1954) 339–341.

[3] T. Aykin, Optimal shift scheduling with multiple break windows, Management Science 42 (1996) 591–602.

[4] S. Bechtold, L. W. Jacobs, Implicit modeling of flexible break assignments in optimal shift scheduling, Management Science 36 (1990) 1339–1351.

[5] J.-R. Birge, F. Louveaux, Introduction to Stochastic Programming, Springer, 1997.

[6] J. F. Bard, D. P. Morton, Y. M. Wang, Workforce planning at USPS mail processing and distribution centers using stochastic optimization, Annals of Operations Research 155 (2007) 51–78.

[7] R. Van Slyke, R. Wets, L-shaped linear programs with applications to optimal control and stochastic programming, SIAM J. Appl. Math. 17 (4) (1969) 638–663.

[8] J.-F. Benders, Partitioning procedures for solving mixed-variables programming problems, Numerische Mathematik 4 (1962) 238–252.

[9] A. Legrain, Génération de scénarios pour la demande en personnels durant plusieurs périodes, Mémoire de maîtrise, École Polytechnique de Montréal (2011).

[10] M. Rekik, J. Cordeau, F. Soumis, Implicit shift scheduling with multiple breaks and work stretch duration restrictions, Journal of Scheduling 13 (1) (2010) 49–75.

[11] H. Sherali, X. Zhu, Advances in Applied Mathematics and Global Optimization, Springer Science, 2009, Ch. 12: Two-Stage Stochastic Mixed-Integer Programs: Algorithms and Insights.

[12] M. Hamdouni, G. Desaulniers, F. Soumis, Parking buses in a depot using block patterns: A Benders decomposition approach for minimizing type mismatches, Computers and Operations Research 34 (11) (2007) 3362–3379.