Scheduling Internal Audit Activities: A Stochastic Combinatorial Optimization Problem⋆

Roberto Rossi[1,2]**, S. Armagan Tarim**[3]**, Brahim Hnich**[4]**, Steven Prestwich**[1]**, Semra Karacaer**[3]

[1] Cork Constraint Computation Centre, University College, Cork, Ireland, e-mail: `{rrossi,s.prestwich}@4c.ucc.ie`
[2] Centre for Telecommunication Value-Chain Driven Research, Ireland, e-mail: `rrossi@4c.ucc.ie`
[3] Department of Management, Hacettepe University, Ankara, Turkey, e-mail: `{armagan.tarim,semra}@hacettepe.edu.tr`
[4] Faculty of Computer Science, Izmir University of Economics, Izmir, Turkey, e-mail: `brahim.hnich@ieu.edu.tr`

Abstract The problem of finding the optimal timing of audit activities within an organisation has been addressed by many researchers. We propose a stochastic programming formulation with Mixed Integer Linear Programming (MILP) and Constraint Programming (CP) certainty-equivalent models. In experiments neither approach dominates the other. However, the CP approach is orders of magnitude faster for large audit times, and almost as fast as the MILP approach for small audit times. This work generalises a previous approach by relaxing the assumption of instantaneous audits, and by prohibiting concurrent auditing.

1 Introduction

Based on costs and benefits that change over time, the focus of the internal audit scheduling problem is how often to conduct an internal audit on an auditable unit. Auditable units are the units upon which internal control procedures are applied, in order to safeguard assets and assure the reliability of information flows. The scope of auditable units depends on organizational characteristics: they could be organizational units (finance, accounting department), geographic regions (branches, cities) or activities (budgeting, purchasing, etc) [4].

The problem of finding the optimal timing of audit activities within an organisation has been addressed by many researchers including Wilson and Ranson [34], Hughes [16], Boritz and Broca [4], and Knechel and Benson [21]. The first study of audit scheduling was by Wilson and Ranson [34] who found the audit frequency that minimizes the discounted present value of losses and audit costs. Audit costs are assumed to be incurred at a uniform rate, while losses in the absence of auditing are assumed to rise exponentially from zero to an asymptotic level. After an audit is conducted, the losses drop to zero but start to accrue until the next audit. In [16] the audits are chosen at the beginning of each of an infinite number of periods, conditional upon available information concerning the state of internal control system, and a model is proposed for determining the optimal timing of internal audits. The model proposed by Boritz and Broca [4] determines the optimal audit interval, assuming that expected losses accrue if a unit remains unaudited and audit cost is incurred each time the decision to audit is made. In order to use their formulation, each audit unit is assessed using an index of loss riskiness called the Audit Unit Priority Score (AUPS). The parameters of the model (i.e. the shape of the loss function and the rate of increase in losses) are determined through auditors' judgmental process, which gives auditors flexibility in the scheduling of audit activities.

In a recent paper by Tarim et al. [30] a stochastic version of the internal audit scheduling problem is formulated under relatively relaxed assumptions. Unlike previous models that determine optimal timing for one audit unit, their model determines the optimal timing of audit activities for multiple audit units. This is important because many firms have more than one auditable unit to which audit resources must be allocated. Moreover, their formulation takes into account uncertainty in the losses accrued in the absence of auditing, and employs a chance-constraint to keep the expected losses below a certain level with a given probability. However, in [30] it is assumed that audit activities are instantaneous, i.e. that conducting an audit does not take any time. The computational issues are not addressed by Tarim et al.

Mixed Integer Linear Programming (MILP) [23] and Constraint Programming (CP) [12] are two orthogonal approaches used to address combinatorial problems. MILP-based methods are rooted at the area of Operations Research [23], whereas CP-based methods are the result of research

by the Artificial Intelligence community in the areas of Logic Programming and Constraint Satisfaction [8,14,31]. MILP and CP methods have both been successfully applied to solve diverse problems such as network synthesis, crew scheduling, planning, and capital budgeting. Determining which type of problems or instances are best solved by which method is an active research area. This is the line of research pursued in this paper. We relax the assumption of instantaneous audits and propose a stochastic programming formulation for this important class of combinatorial problems involving uncertainty. To solve this stochastic program we develop two alternatives deterministic equivalent models: a MILP model and a CP model. The two approaches are complementary in the sense that for some instances MILP is superior and for other instances CP is superior. Our numerical experiments show that the certainty-equivalent MILP formulation is efficient when the time to perform an audit is relatively short. However, as the audit time gets longer our CP model proved to be significantly much more effective than the MILP model.

The paper is organised as follows: Section 2 describes the problem, Section 3 provides a stochastic programming formulation of the problem, Section 4 surveys possible solution methods, Section 5 reports experimental results comparing methods, and Section 6 concludes the paper. Finally, in the Appendix we provide a complete list of the notation adopted in the paper.

2 Problem statement

We consider a planning horizon comprising N time periods. We are given a set of M audit units over which random losses may accrue over time. In particular, l_t^m corresponds to the losses that accrue in audit unit m during period t. l_t^m is a random variable with a known probability density function $g_{l_t^m}(l_t^m)$. For convenience, losses in each period are assumed to be normally distributed with a constant coefficient of variation: $\rho = \sigma_t^m/\mu_t^m$ in this problem, but this assumption may be relaxed without loss of generality. The distribution of losses may vary from period to period, i.e. it is non-stationary. Losses in different time periods are assumed to be independent. Figure 1 illustrates expected losses on a single auditable unit.

Without loss of generality, we consider the case in which a single audit team has to be employed to keep losses under control. Auditing is a time-consuming task, and we assume that the team is given a strict deadline for performing an audit. Specifically, an audit must be completed in T time periods ($T > 0$). Therefore after T periods the accrued losses will drop to zero. If a team has already started auditing a unit at a given period, then no other audit can be initiated during this period for the given audit team. Figure 2 depicts such a situation in which an audit duration of 2 periods is assumed. An audit scheduled for unit 1 at the beginning of period 3 rules out any following audit for unit 2 until period 5.
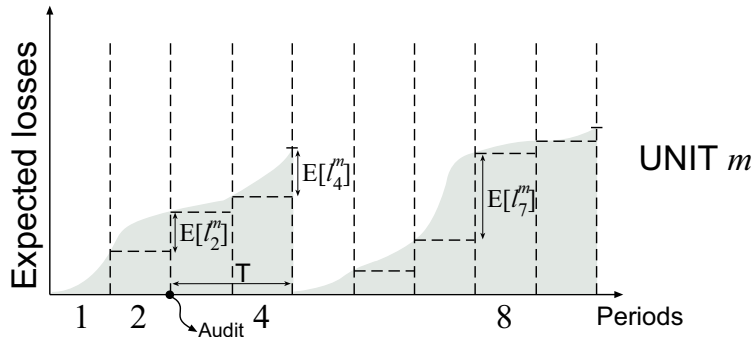
Fig. 1 Expected losses; $E[l_t^m]$ denotes the expected value of l_t^m

Fig. 2 Multiple units

Note that the timing of audits are fixed once and for all at the beginning of the planning horizon, and cannot be changed thereafter even if it is suspected that certain auditable units have accrued unexpected losses. The objective is to find the optimal audit schedule while respecting the maximum loss level criterion. That is, the invariant audit costs (i.e. fixed audit costs incurred each time an audit is conducted) and expected total discounted audit losses (i.e. cumulative losses accrued at the end of each

period) are minimized by satisfying a maximum loss level constraint, which in this problem is defined by specifying a minimum probability α that the losses will not exceed a predetermined level $\bar{L}$ (allowed maximum loss) in any given audit period for any auditable unit.

Example 1 In what follows we will employ a running example to better exemplify the above concepts. We consider the following simple instance:

M : the total number of audit units, equal to 2,
N : the number of periods in the planning horizon, equal to 6,
T : the duration of an audit in time periods, equal to 2,
a : the fixed cost incurred each time an audit is conducted,
       equal to 100
h : the loss discount factor measuring the opportunity cost associated
       to a given loss level, equal to 1.
$\bar{L}$: a threshold indicating the maximum allowed loss level in each ,
       period, equal to 200,
α : the probability of not exceeding the loss threshold $\bar{L}$, equal to 0.95.

We assume the losses accrued in each period to be normally distributed with a constant coefficient of variation $\rho = 0.2$, where $\rho = \sigma_t^m / \mu_t^m$. The expected value, μ_t^m, for the losses in each period t and for each audit unit m is respectively $\{50, 30, 50, 30, 50, 30\}$ in each period $t = 1, \ldots, 6$ for audit unit 1, and $\{10, 20, 30, 40, 50, 60\}$ in each period for audit unit 2.

3 Stochastic programming formulation

Stochastic programming [2] is a well known modeling technique that deals with problems where uncertainty comes into play. Problems of optimization under uncertainty are characterized by the necessity of making decisions without knowing what their full effect will be. Stochastic programming needs to represent uncertain elements of the problem. Typically random variables are employed to model this uncertainty to which probability theory [32,20] can be applied. For this purpose such uncertain elements must have a known probability distribution. The typical requirement in stochastic programs is to maintain certain constraints, called *chance constraints* [6], satisfied at a prescribed level of probability. The objective is typically related to the minimization/maximization of some expectation on the problem costs.

    The stochastic programming model we propose balances the discounted cost of losses accrued due to lack of audits with the cost of conducting audits. Taking the discounted cost of losses into account is particularly relevant when the cost of money must be considered. Consider, for instance, the situation in which some losses are due to a specific reason in a given period. Then in the following periods this loss will affect company assets until the originating factor is discovered by an audit and cleared. Obviously

these effects will have a higher impact the longer it takes to clear such an originating factor. Consider, for instance, the case in which a company's tax liabilities are overestimated. The capital tied into tax liabilities could be invested in a more profitable way if the accounts were not flawed. In this case the discount factor would reflect the opportunity cost associated with the fact that capitals may be invested in a more profitable way if an audit were scheduled.

We employ the expected value criterion to minimize the sum of the expected discounted period losses and audit costs over an N period planning horizon. Let us consider, without loss of generality, an initial loss levels L_1^m set to any non-negative values for each audit unit $m = 1, \ldots, M$. Let

L_t^m : the loss level in audit unit m at the beginning of period t.

The objective function below and the following constraints give the optimum audit timing for each audit unit by minimizing $E[TC]$, that is the sum of expected audit costs and discounted period losses that are expected to accrue in the absence of auditing.

$$\min E[TC] = \sum_{m=1}^{M} \int_{l_1^m} \int_{l_2^m} \cdots \int_{l_N^m} \sum_{t=1}^{N} (aK_t^m + h(L_t^m + l_t^m)) \qquad (1)$$
$$g_{l_1^m}(l_1^m) g_{l_2^m}(l_2^m) \ldots g_{l_N^m}(l_N^m) \mathrm{d}(l_1^m) d(l_2^m) \ldots d(l_N^m)$$

where

M : the total number of audit units,
N : the number of periods in the planning horizon,
a : the amount of cost incurred each time an audit is conducted,
h : the loss discount factor measuring the opportunity cost associated
         to a given loss level.
K_t^m : a variable that takes the value of 1 if an internal audit
         (lasting T periods) is started for audit unit m in period t, otherwise 0.

The above objective function is subject to several constraints. If $(K_t^m = 1)$ an internal audit is conducted at the beginning of period t (i.e. at the end of period $t - 1$), then the loss level at the beginning of period $t + T$ should be 0. Yet, if an internal audit is not conducted, the loss level at the beginning of period $t + T$ will be equal to the loss level at the beginning of the preceding period plus the loss accrued during the preceding period. This can be expressed as

$$L_{t+T}^m \geq L_{t+T-1}^m + l_{t+T-1}^m - \mathfrak{M}K_t^m, \quad m = 1, \ldots, M, \quad t = 1, \ldots, N-T, \ (2)$$

where $\mathfrak{M}$ is some very large number. Obviously in the first T periods no audit can be completed, therefore
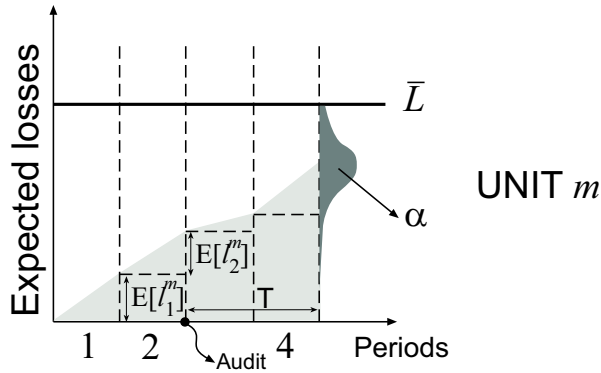
$$L_t^m \geq L_{t-1}^m + l_{t-1}^m, \quad m = 1, \ldots, M, \quad t = 1, \ldots, T. \qquad (3)$$

For convenience we consider $l_t^m = L_t^m = 0$, for $\{t | t < 1\}$.

Now consider a plan for audit unit m, which schedules r audits over the N period planning horizon with audits conducted at $\{A_1^m, \ldots, A_r^m\}$, where, $A_j^m > A_{j-1}^m$, $A_r^m \leq N - T$. For convenience $A_1^m = 1 - T$, because the initial loss level is set to 0; $A_{r+1}^m = N - T + 1$ is defined as the earliest period for which an associated audit would be completed only after the end of the horizon. The associated audits will take place at the beginning of periods A_i^m, $i = 1$ to r. In the considered plan there are clearly no audits scheduled for audit unit m except at periods $A_1^m, \ldots, A_r^m$. The accumulated loss level L_{t+1}^m carried over from period t to period $t+1$ is the loss to date since the last completed audit. This can be written as

$$L_{t+1}^m = \sum_{k=A_i^m+T}^{t} l_k^m, \quad A_i^m + T \leq t < A_{i+1}^m + T, \quad i = 1, \ldots, r. \quad (4)$$

As defined above, α is the desired minimum probability that the loss level in any time period will not exceed a subjectively determined level, $\bar{L}$ (Figure 3). In this regard the chance constraint becomes

Fig. 3 Chance-constraint on the maximum loss level. Assuming losses to be normally distributed. α is the desired minimum probability (area marked in the figure) that the loss level in any time period will not exceed a subjectively determined level, $\bar{L}$

$$\Pr\{L_t^m + l_t^m \leq \bar{L}\} \geq \alpha, \quad t = 1 + T, \ldots, N. \quad (5)$$

Using Eq.4, this can be written alternatively as, for $t \geq 1 + T$

$$\Pr\{\sum_{k=A_i^m+T}^{t} l_k^m \leq \bar{L}\} \geq \alpha, \quad A_i^m + T \leq t < A_{i+1}^m + T, \quad i = 1, \ldots, r, \quad (6)$$

which implies,

$$G_{l_{A_i^m+T}^m + l_{A_i^m+T+1}^m + \ldots + l_t^m}(\bar{L}) \geq \alpha, \quad A_i^m + T \leq t < A_{i+1}^m + T, \quad (7)$$

where $G_{l_t^m}(x) = \int_{-\infty}^x g_{l_t^m}(\tau)\mathrm{d}\tau$ is the cumulative distribution function of l_t^m. By assuming $G_{l_t^m}(x)$ to be strictly increasing, thus invertible, Eq. 7 can then be rewritten as

$$\bar{L} \geq G^{-1}_{l^m_{A_i^m+T}+l^m_{A_i^m+T+1}+\ldots+l_t^m}(\alpha) \quad A_i^m + T \leq t < A_{i+1}^m + T, \qquad (8)$$

where $G^{-1}_{l_t^m}(\alpha)$ is the inverse cumulative distribution function (or α-quantile) of l_t^m.

Since the problem has a finite planning horizon of N periods, for all the relevant cases the right-hand side of Eq.8, $G^{-1}_{l^m_{A_i^m+T}+l^m_{A_i^m+T+1}+\ldots+l_t^m}(\alpha)$, can be computed or possibly read from a table, once the form of $g_{l_t^m}(.)$ is decided. If the binary variable $P_{t,j}^m$ is defined as taking a value of 1 if the most recent audit prior to period t was in period j and zero elsewhere for a given audit unit m, then Eq.8 can be written as

$$\bar{L} \geq \sum_{j=1}^t \left(G^{-1}_{l_j^m+l_{j+1}^m+\ldots+l_t^m}(\alpha)\ P_{t,j-T}^m \right) \qquad (9)$$

There can be at most only one most recent audit prior to period t. Thus $P_{t,j}^m$ must satisfy

$$\sum_{j=1-T}^t P_{t,j}^m = 1, \quad m = 1,\ldots,M, \quad t = 1,\ldots,N. \qquad (10)$$

Note that losses in the first T periods cannot be controlled with respect to the threshold $\bar{L}$ and the probability α. In fact benefits from the first possible audit appear only in period $T + 1$. Therefore we assume, for all $m = 1,\ldots,M$, $K_{1-T}^m = 1$ (according to the fact that initial losses should be equal to 0) and $K_{i-T}^m = 0$, $i = 2,\ldots,T$.

The following Eq. 11 is necessary to identify uniquely the period in which the most recent audit prior to any period t took place. For each audit unit $m = 1,\ldots,M$

$$P_{t,j}^m \geq K_j^m - \sum_{k=j+1}^{t-T} K_k^m, \quad t = 1,\ldots,N, \quad j = 1-T,\ldots,t-T. \qquad (11)$$

It is not common practice for internal audit teams to conduct multiple audits simultaneously. To have a modicum of resemblance to reality, as already stated, here it is assumed that a team can conduct an internal audit only for one audit unit at a given time period. In our model we shall consider the following capacity constraint

$$\sum_{k=1}^m K_t^m \leq C, \quad m = 1,\ldots,M, \quad t = 1,\ldots,N-T, \qquad (12)$$

which states that the firm can assign at most C audit teams to conduct audits in any given time period. For simplicity, in what follows we will assume $C = 1$.
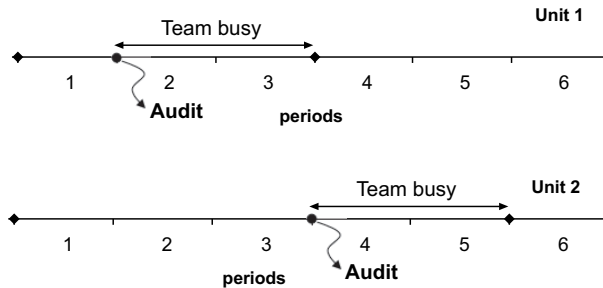
Example 2 For the running example introduced in Section 2, in Table 1 we show the values of $G^{-1}_{l^m_j+l^m_{j+1}+\ldots+l^m_t}(\alpha)$ for audit unit $m = 1, 2$. Values in bold are those that satisfy Constraint 9, that is values that stay below the loss threshold $\bar{L} = 200$. Underlined values identify audit cycles[1] in the optimal solution. The optimal audit plan is also presented graphically in Fig 4. In

Unit 1

	t					
	1	2	3	4	5	6
j 1	66.5	99.2	**155.3**	187.1	241.7	273.2
2		39.9	99.2	131.6	187.1	218.9
3			66.5	99.2	155.3	187.1
4				39.9	99.2	**131.6**
5					66.5	99.2
6						39.9

Unit 2

	t					
	1	2	3	4	5	6
j 1	13.3	37.4	72.3	118.0	**174.4**	241.4
2		26.6	61.9	107.7	164.2	231.2
3			39.9	86.5	143.3	210.5
4				53.2	111.1	178.9
5					66.5	135.7
6						**79.7**

Table 1 Values of $G^{-1}_{l^m_j+l^m_{j+1}+\ldots+l^m_t}(\alpha)$ for audit unit $m = 1, 2$. Values in bold are those that satisfy Constraint 9. That is values that stay below the loss threshold $\bar{L} = 200$. Underlined values identify audit cycles in the optimal solution.

Fig. 4 Optimal audit plan for the numerical example

this plan, in order to keep accrued losses under control, one single audit for unit 1 is scheduled at the beginning of period 2, and this audit terminates

---

[1] An audit cycle is a set of periods $\{j, \ldots, t\}$, $j \leq t$, where j is the first period after the completion of an audit, and no other audit is completed by period t.

at the end of period 3. Similarly, for unit 2 a single audit is scheduled at
period 4 and this audit terminates at the end of period 5. The expected
total cost for this plan is 1090.

We now show in detail how the values in Table 1 are computed. Let
G_N be the standard normal distribution function. This function is strictly
increasing, therefore G_N^{-1} is uniquely defined[2]. $G_N^{-1}(0.95) = 1.645$ corre-
sponds to the 0.95-quantile. Therefore, since all the random variables l_t^m,
$t = 1, \ldots, N$, $m = 1, \ldots, M$, are independent and normally distributed,
$G_{l_2^1 + l_3^1 + l_4^1}^{-1}(0.95) = 1.645 * 0.2 * \sqrt{30^2 + 50^2 + 30^2} = 131.6$. This value can be
found in the first matrix presented in Table 1 at position (2,4). It corre-
sponds to a sequence of periods starting in period 2 and ending in period 4,
where no audit is completed and where the last audit performed has been
completed by the end of period 1 (therefore losses at the beginning of period
2 are null). Since $131.6 < 200$ it follows that this set of periods constitutes
a feasible audit cycle.

4 Solution methods

In this section we present two alternative certainty equivalent [2] models for
the stochastic programming model presented in the former section: a Mixed
Integer Linear Programming model and a Constraint Programming model.

4.1 A certainty equivalent MILP model

In Linear Programming [10,7,27] a model, called a "program", consists of
continuous variables and linear constraints (inequalities or equalities), and
the aim is to optimize(minimize or maximize) a linear cost function. In ma-
trix notation the standard form of a linear program is $\min\{c^T x | Ax = b, x \geq 0\}$, where $c \in \mathbb{R}^n$, $b \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$. Here c represents the cost vector
and x is the vector of variables. Linear programs are usually solved by the
simplex method [9] which is very fast in practice, though it has exponential
worst-case time complexity. A Mixed Integer Linear Program (MILP) is a
Linear Program (LP) plus integrality requirements on some decision vari-
ables. Discrete variables in MILP are often 0-1 variables representing yes/no,
on/off or true/false choices. Logical conditions between 0-1 variables such
as $\vee$ (or), $\wedge$ (and), $\neg$ (not), $\Rightarrow$ (if … then), and $\Leftrightarrow$ (if and only if) can
be imposed using appropriate linear constraints [33]. Solution methods for
MILP typically employ tree search in which internal nodes correspond to
partial solutions, branches are choices partitioning the search space, and
leaf nodes are solutions. Branching is intertwined with a relaxation to elim-
inate the exploration of nodes for which the relaxation is either infeasible
or worse than the best solution found so far. Each node represents a partial

---

[2] Tables are available for obtaining values of the inverse normal cumulative
distribution function (also known as α-*quantile* [32])

assignment of the discrete variables, and at each node a relaxation is formed by turning the integrality requirements into bounds, thus transforming the subproblem into an LP. This LP is solved, and if the solution is not sub-optimal then descendant nodes are formed by branching on the fractional relaxation value of a discrete variable. The historical popularity of MILP derives from Dantzig's discovery that the vocabulary of LP is surprisingly versatile in many applications. MILP-based methods have been developed over the last four decades by the Operations Research community [23].

The mathematical programming model of the previous section, as presented, is a stochastic nonlinear combinatorial optimization model, which is extremely complex to solve. In this section we adopt the static-dynamic uncertainty strategy, proposed by Bookbinder and Tan [3] to solve their stochastic inventory lot-sizing problem, and apply it to the mathematical programming model of Section 3. The model can be expressed as minimizing the objective function given in Eq.1 subject to the constraints Eqs. (2-4), (9-12), and non-negativity and $0/1$ integrality conditions for L_t^m, K_t^m and $P_{i,j}^m$.

In our internal audit scheduling problem the analysis is completed at the beginning of the planning horizon by taking expectations (see [3]). Hence the deterministic equivalent model for the original chance-constrained stochastic programming model is obtained. The resultant model is in the form of a mixed-integer program, given below, in which the expected value operator is denoted by $E[.]$.

$$\min \quad \sum_{m=1}^{M} \left(\sum_{t=1}^{N} a K_t^m + \sum_{t=1}^{N} h E[L_t^m + l_t^m] \right) \tag{13}$$

$$\text{subject to, for } m = 1, \ldots, M \tag{14}$$

$$E[L_1^m] = 0, \tag{15}$$

$$E[L_{t+1}^m] \geq E[L_t^m] + E[l_t^m], \quad t = 1, \ldots, T \tag{16}$$

$$E[L_{t+T}^m] \geq E[L_{t-1+T}^m] + E[l_{t-1+T}^m] - \mathfrak{M} K_t^m, \quad t = 1, \ldots, N - T \tag{17}$$

$$\sum_{k=1}^{M} \sum_{h=1}^{\min(t+T-1,N)} K_h^k \leq 1, \quad t = 1, \ldots, N \tag{18}$$

$$\bar{L} \geq \sum_{j=1}^{t} G^{-1}_{l_j^m + l_{j+1}^m + \ldots + l_t^m}(\alpha) \cdot P_{t,j-T}^m, \quad t = 1, \ldots, N \tag{19}$$

$$\sum_{j=1}^{t} P_{t,j}^m = 1, \quad t = 1 - T, \ldots, N \tag{20}$$

$$P_{m,t,j} \geq K_j^m - \sum_{k=j+1}^{t-T} K_k^m, \quad j = 1 - T, \ldots, t - T \tag{21}$$

$$K_{1-T}^m = 1, \tag{22}$$

$$K_{t-T}^m = 0, \quad t = 2, \ldots, T \tag{23}$$

$$E[L_t^m] \geq 0 \tag{24}$$

$$K_t^m, P_{t,j}^m \in \{0,1\}, \quad t = 1, \ldots, N, \quad j = 1 - T, \ldots, t \tag{25}$$

This model thus determines the optimal audit schedule by balancing the fixed audit costs and discounted expected period losses that accrue in

the absence of auditing. The problem is to determine the values of the
0/1 integer variables, K_t^m for $m = 1, \ldots, M$, $t = 1, \ldots, N$, and $P_{t,j}^m$ for
$t = 1, \ldots, N$, $j = 1 - T, \ldots, t$, and the non-negative continuous variable
$E[L_t^m]$ for $t = 1, \ldots, N$, that minimize the objective function. The times
of the audit reviews in each audit unit m are given by the values of t such
that $K_t^m = 1$. Constraint 15 states that the initial losses are equal to 0.
Constraint 16 lets expected losses accumulate in the first T periods for each
audit unit, since no audit can be terminated before period $T+1$. Constraint
17 states that if an audit is planned in period t, then expected losses must
drop to zero in period $t+T$, as soon as the audit terminates, while if no audit
is planned in period t, then expected losses in period $t + T$ must be equal
to the expected losses accumulate till the beginning of the previous period
$(t + T - 1)$ plus the expected losses accrued in such a period. Constraint
18 prevents multiple audits in any given period. If an audit team starts
an audit in period t on a given unit, this means that no other audit can
be performed on any unit before period $t + T$. Constraints 19, 20 and 21
implement Eq. 8 and therefore they identify feasible audit schedules, that
is those for which losses never exceed the given threshold $\bar{L}$ more than α
percent of the times.

Example 3 By employing the mathematical programming model presented
in this section we can solve the running example originally presented in Sec-
tion 2. The optimal plan, which we already described in Section 3, is shown
in Fig. 5. In this picture, we also show the expected losses accumulated
in each period and computed by the MILP model. The plan schedules one
single audit for unit 1 at period 2, and for unit 2 at period 4. The expected
total cost for this plan is 1090. Note that for each audit, losses drop to 0
only after $T = 2$ periods, which is in fact the time required to perform
an audit. It is also clear from the plan shown that the audit team cannot
perform multiple audits at any given time period.

4.2 A certainty equivalent CP model

A *Constraint Satisfaction Problem* (CSP) [1,5,22] is a triple $\langle V, C, D \rangle$ where
V is a set of decision variables, D is a function mapping each element of V
to a domain of potential values, and C is a set of constraints stating allowed
combinations of values for subsets of variables in V. A *solution* to a CSP
is simply a set of values of the variables such that the values are in the
domains of the variables and all the constraints are satisfied. We may also
be interested in finding a feasible solution that minimizes or maximizes the
value of a given objective function over a subset of the variables.

We now recall some key concepts in *Constraint Programming* (CP): con-
straint filtering algorithm, constraint propagation and arc-consistency [26].
A *filtering algorithm* is typically associated with a constraint, and removes
values from the domains of its variables that cannot belong to any solution
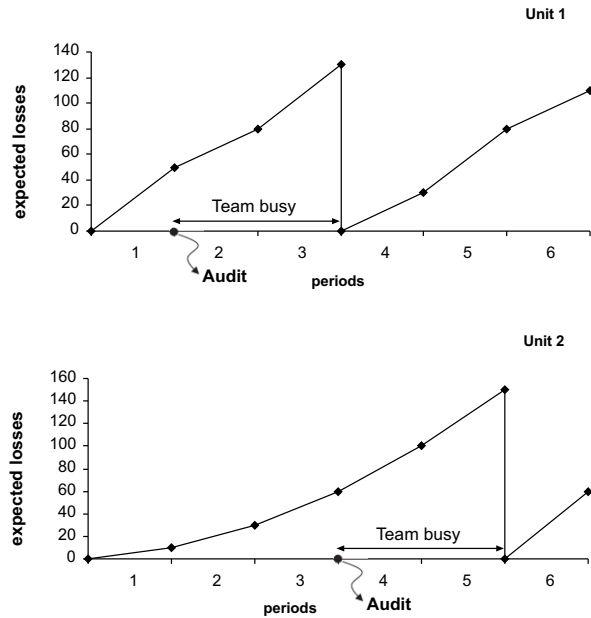of the CSP. These algorithms are repeatedly called until no new deduction

Fig. 5 Optimal audit plan for the numerical example

can be made, a process called *propagation*. In conjunction with this process
CP uses a search procedure (typically a backtracking algorithm) in which
filtering is systematically applied whenever the domain of a variable is mod-
ified. One of the most interesting properties of a filtering algorithm is *arc-
consistency*: we say that a filtering algorithm associated with a constraint
establishes arc-consistency if it removes all the values from the domains of
the variables involved in the constraint that are not consistent with the con-
straint. Studies on arc-consistency are often limited to binary constraints,
but modeling problems by means of binary constraints has drawbacks: they
are not very expressive, and their domain reduction is typically weak. To
overcome both these drawbacks, constraints that capture a relation among
a non-fixed number of variables were introduced. These constraints are not
only more expressive than the equivalent aggregation of simple constraints,
but they can be associated with more powerful filtering algorithms that
take into account the simultaneous presence of several simple constraints
to further reduce the domains of the variables. These more powerful con-
straints are called *global constraints*. One of the best-known examples is the
`alldiff` constraint [25], both because of its expressiveness and its efficiency
in establishing arc-consistency. A comprehensive and up-to-date survey of
the state of knowledge regarding CP is Apt [1], while a general account

of the CP–MP relationship is given by Brailsford et al. [5] and Lustig and Puget [22].

In this section we propose a CP reformulation for the mathematical programming model in Section 4.1. This reformulation follows the approach originally proposed in [29], it exploits non-binary (global) constraints and other features of CP. The CP model, similarly to the mathematical programming one, can be expressed as minimizing the objective function given in Eq.1. But, as we shall see, in the CP model constraints Eqs. (2-4) and (9-12) are now reformulated and expressed in a more compact and readable way. Furthermore, as we will see, the number of decision variables employed is dramatically reduced as we do not employ anymore the binary decision variables $P_{i,j}^m$. The number of constraints is also significantly reduced. The CP model is as follows.

$$\min \quad \sum_{m=1}^{M} \left(\sum_{t=1}^{N} aK_t^m + \sum_{t=1}^{N} hE[L_t^m + l_t^m] \right) \tag{26}$$

subject to, for $m = 1, \ldots, M$ (27)

$$E[L_1^m] = 0, \tag{28}$$

$$E[L_{t+1}^m] \geq E[L_t^m] + E[l_t^m], \quad t = 1, \ldots, T \tag{29}$$

$$K_t^m = 1 \rightarrow E[L_{t+T}^m] = 0, \quad t = 1, \ldots, N - T \tag{30}$$

$$K_t^m = 0 \rightarrow E[L_{t+T}^m] = E[L_{t-1+T}^m] + E[l_{t-1+T}^m], \quad t = 1, \ldots, N - T \tag{31}$$

$$\sum_{k=1}^{M} \sum_{h=1}^{\min(t+T-1,N)} K_h^k \leq 1, \quad t = 1, \ldots, N \tag{32}$$

$$\Phi\left[m, t + T, \max\left(1, \max_{j=1,\ldots,t} (j+T) \cdot K_j^m \right) \right] \geq 0, \quad t = 1 - T, \ldots, N - T \tag{33}$$

where

$$\Phi[m, t, j] = \bar{L} - G_{l_j^m + l_{j+1}^m + \ldots + l_t^m}^{-1}(\alpha), \quad t = 1, \ldots, N, \quad j = 1, \ldots, N.$$

The objective function, as in the mathematical programming model, balances the fixed audit costs and discounted expected period losses that accrue in the absence of auditing. Constraint 28 states that the initial losses are equal to 0. Constraint 29 lets losses accumulate in the first T periods for each audit unit, since no audit can be terminated before period $T + 1$. Constraint 30 states that if an audit is planned in period t, then losses must drop to zero in period $t + T$, as soon as the audit terminates. Conversely, Constraint 31 states that if no audit is planned in period t, then losses in period $t+T$ must be equal to the losses accumulated till the beginning of the previous period $(t + T - 1)$ plus the losses accrued in such a period. These two non-linear constraints are equivalent to Constraint 17 in the mathematical programming formulation. Constraint 32, similarly to Constraint 18, prevents multiple audits in any given period. If an audit team starts an audit in period t on a given unit, this means that no other audit can be performed on any unit before period $t + T$. Constraint 33 identifies feasible audit schedules, that is those for which losses never exceed the given threshold $\bar{L}$ more than α percent of the time. This constraint replaces the

set of Constraints 19, 20 and 21. The model given in Eqs. 26–33 can be directly implemented using the OPL optimization programming language [15]. It should be noted that, in OPL, Constraint 33 is implemented using the `element`(I, A, J) constraint [13]. The `element` constraint holds iff $A[I] = J$, where I and J are decision variables, and A is an array of decision variables.

5 Experiments

In this section, we compare the computational performance of the MILP formulation, presented in Section 4.1, versus the CP equivalent formulation, presented in Section 4.2, on a number of test problems.

Computational tests are performed on a 1.5GHz, 2GB RAM, Centrino machine using ILOG Cplex 9.0 [17] in OPL Studio 3.7 [18]. The packages are used with their default settings.

In the MILP model, the $\mathfrak{M}$ in Constraint 17 must have a numerical value. It is well known that the computational performance of the MILP model can be improved by choosing $\mathfrak{M}$ as small as possible, without ruling out any possible solution. It is also clear that in different time periods, the corresponding $\mathfrak{M}$ may have been assigned different numerical values. One way of generating such $\mathfrak{M}$ is by observing that, by assuming a reasonably high service level (that is $\alpha > 0.5$) the loss level will never exceed $\bar{L}$. Hence, $\mathfrak{M} = \bar{L}$.

It should be also noted that, while the integer program is treated in its matrix form, and different heuristics are used to choose the variable to branch on based on the solution of the LP relaxation that is solved at each node, in a CP approach the user specifies the branching strategy in terms of the formulation of the problem. The following search strategy is employed in solving the CP model proposed: $K_t^m = 0$ and 1 are tried in order, for all $m \in \{1, \ldots, M\}$, for all $t \in \{1, \ldots, N\}$.

5.1 Experimental Settings

The design of the test problems is as follows. We consider the following inputs:

M : the total number of audit units, equal to 5,

N : the number of periods in the planning horizon, taking values in $\{20, 30, 40\}$,

T : the duration of an audit in time periods, taking values in $\{1, \ldots, 6\}$,

a : the amount of cost incurred each time an audit is conducted, taking values in $\{500, 750, 1000\}$

h : the loss discount factor measuring the opportunity cost associated

       to a given loss level, equal to 1.

$\bar{L}$: a threshold indicating the maximum allowed loss level in each ,
       period taking values in $\{1500, 2500, 3500\}$,

α : the probability of not exceeding the loss threshold $\bar{L}$, equal to 0.95.

We assume the losses accrued in each period to be normally distributed with a constant coefficient of variation $\rho \in \{0.15, 0.3\}$, where $\rho = \sigma_t^m / \mu_t^m$. The expected value μ_t^m for the losses in each period t and for each audit unit m is shown in Fig. 6.

The total number of test problems generated is 108. We further partition our set of problem instances into two classes as follows:
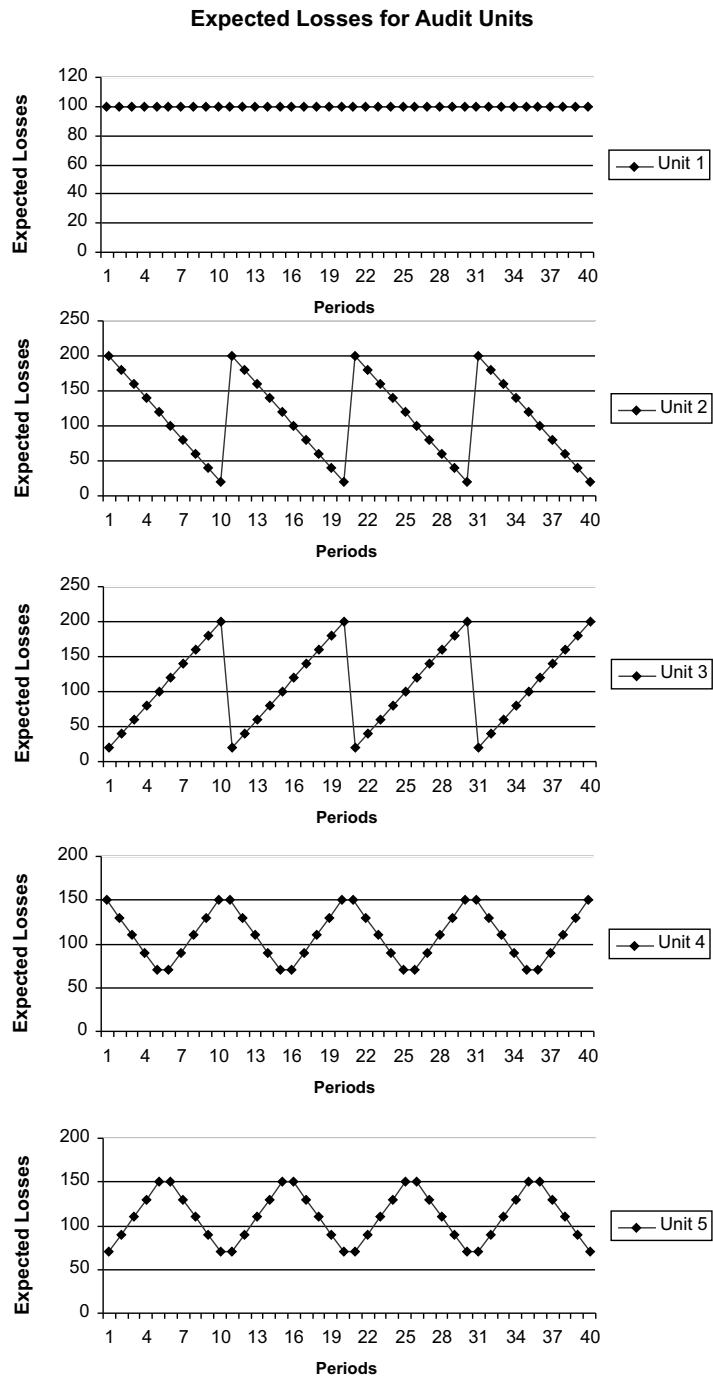
- The instances where the audit time is 1, i.e., $T = 1$ (18 instances);
- The instances where the audit time is greater than 1, i.e., $T \in \{2, \ldots, 6\}$ (90 instances).

We now analyze each set separately.

5.1.1 Instances for which $T = 1$ For each test problem the solution time (in seconds), for both the MILP and the CP approach, is given in Table 2. In this table bold figures highlight the approach that produced the best run time. In Table 3 instead we reported for the MILP approach and for the CP approach, respectively, the simplex iterations performed and the nodes explored. In this first set of 18 instances, where $T = 1$, the MILP approach always dominates the CP approach in terms of run time. Nevertheless the discrepancy between the two approaches reaches only one order of magnitude in the worst case. In the average case MILP is faster than CP by a factor of 7.7.

5.1.2 Instances for which $T > 1$ For each test problem the solution time (in seconds), for both the MILP and the CP approach, is given in Table 4. In this table, again bold figures highlight the approach that produced the best run time, while those instances for which the figures are underlined are infeasible. In Table 5 we reported for the MILP approach and for the CP approach, respectively, the simplex iterations performed and the nodes explored.

Expected Losses for Audit Units

Fig. 6 Expected value μ_t^m for the losses in each period t and for each audit unit m

$\bar{L}$		1500			2500			3500				
a	500	750	1000	500	750	1000	500	750	1000			
$\rho = 0.15$											T	N
MILP	**1.1**	**0.98**	**0.7**	**1.4**	**1.5**	**0.56**	**0.72**	**1.4**	**0.59**		1	10
CP	6.2	7.6	8.2	6.7	8.1	8.2	7.2	9.2	8.8		1	10
$\rho = 0.3$											T	N
MILP	**1.1**	**0.96**	**0.63**	**1.3**	**1.1**	**0.57**	**1.2**	**1.0**	**0.59**		1	10
CP	5.9	6.8	7.5	6.5	7.5	7.8	6.5	7.7	8.05		1	10

Table 2 Computational times (in Secs) for the MILP approach (MILP) and for the CP approach (CP). Bold figures in the tables highlight the approach that produced the best run time

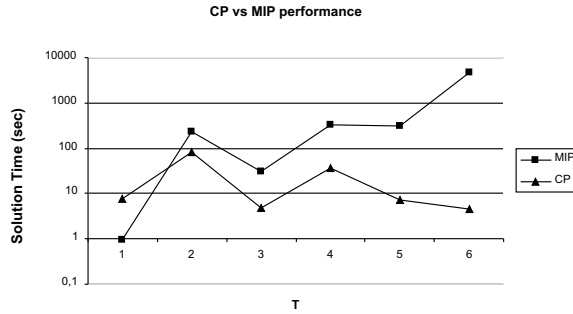| $\bar{L}$		1500			2500			3500				
a	500	750	1000	500	750	1000	500	750	1000			
$\rho = 0.15$											T	N
MILP-SI	**14122**	**13383**	**8071**	**15868**	**13483**	**6666**	**15350**	**13854**	**7083**		1	10
CP-Nod	83480	103442	108266	83480	103442	108266	83480	103372	108266		1	10
$\rho = 0.3$											T	N
MILP-SI	**16084**	**13460**	**8381**	**16937**	**13649**	**6683**	**15357**	**13854**	**7083**		1	10
CP-Nod 1	83480	103442	108266	83480	103442	108266	83480	103442	108266		1	10

Table 3 Simplex iterations performed by the MILP approach (MILP-SI) and Nodes explored by the CP approach (CP-Nod). Bold figures in the tables highlight the approach that produced the best run time

$\bar{L}$		1500			2500			3500			
a	500	750	1000	500	750	1000	500	750	1000		
$\rho = 0.15$										T	N
MILP	150	120	98	240	240	200	310	420	310	2	20
CP	**9.4**	**9.6**	**9.6**	**92**	**110**	**130**	**110**	**120**	**170**	2	20
MILP	22	17	41	37	25	26	36	30	48	3	20
CP	**0.12**	**0.11**	**0.10**	**7.6**	**7.3**	**7.4**	**6.4**	**6.8**	**7.6**	3	20
MILP	20	110	80	500	320	400	610	540	610	4	30
CP	**0.07**	**0.07**	**0.07**	**8.3**	**7.9**	**8.4**	**100**	**100**	**100**	4	30
MILP	**0**	**0**	**0**	700	1500	810	250	140	420	5	30
CP	0.06	0.07	0.06	**1.0**	**1.0**	**1.0**	**20**	**20**	**21**	5	30
MILP	**0**	**0**	**0**	2300	1500	1900	660	1200	1500	6	40
CP	0.11	0.11	0.10	**0.45**	**0.39**	**0.41**	**17**	**20**	**17**	6	40
$\rho = 0.3$										T	N
MILP	70	100	87	190	420	200	340	400	260	2	20
CP	**4.3**	**4.12**	**4.3**	**87**	120	130	**100**	**140**	**130**	2	20
MILP	17	12	36	22	31	26	36	34	33	3	20
CP	**0.08**	**0.08**	**0.08**	**6.2**	**7.2**	**6.9**	**6.5**	**6.8**	**7.4**	3	20
MILP	39	20	23	160	370	550	550	580	620	4	30
CP	**0.06**	**0.06**	**0.06**	**5.0**	**4.9**	**5.2**	**110**	**110**	**120**	4	30
MILP	**0**	**0**	**0**	400	310	320	300	130	440	5	30
CP	0.07	0.07	0.07	**0.77**	**0.77**	**0.77**	**20**	**20**	**20**	5	30
MILP	**0**	**0**	**0**	1600	1500	700	20000	24000	+30000	6	40
CP	0.1	0.1	0.1	**0.35**	**0.39**	**0.47**	**7.7**	**7.7**	**7.7**	6	40

Table 4 Computational times (in Secs) for the MILP approach (MILP) and for the CP approach (CP). Bold figures in the tables highlight the approach that produced the best run time. Underlined figures are infeasible instances. +30000 means that the search has been stopped before infeasibility could be proved, after 30000 secs (8,3 hours)

$\bar{L}$		1500			2500			3500				
a	500	750	1000	500	750	1000	500	750	1000		T	N
$\rho = 0.15$												
MILP-SI	737530	534238	444055	1115993	1102108	952999	1548908	2182999	1346656		2	20
CP-Nod	**70246**	**75317**	**80386**	**636450**	**764147**	**911016**	**636450**	**764147**	**911134**		2	20
MILP-SI	115666	107912	227602	191912	133002	137485	179505	150434	204205		3	20
CP-Nod	**511**	511	**511**	**48025**	**51003**	**54077**	**48025**	**51003**	**54077**		3	20
MILP-SI	23915	176870	122716	1745261	1056964	1285394	1842453	1580697	1565614		4	30
CP-Nod	**90**	90	**90**	**56456**	**56526**	**56620**	**517734**	**538358**	**559576**		4	30
MILP-SI	p.	p.	p.	1889563	4362836	2365598	889056	484259	1493608		5	30
CP-Nod	26	26	26	**6017**	**6017**	**6017**	**104956**	**107308**	**109671**		5	30
MILP-SI	p.	p.	p.	3068784	2184342	2979721	1440912	2280890	3245663		6	40
CP-Nod	17	17	17	**1358**	**1358**	**1358**	**94461**	**94465**	**94467**		6	40
$\rho = 0.3$											T	N
MILP-SI	358921	455248	425701	902365	1946679	965218	1477861	2142193	1347372		2	20
CP-Nod	**38004**	**39394**	**40679**	**636450**	764147	911134	**636450**	**764147**	**911134**		2	20
MILP-SI	105962	80243	210615	117851	174027	129448	181685	165411	177510		3	20
CP-Nod	**511**	**511**	**511**	**48025**	**51003**	**54077**	**48025**	**51003**	**54077**		3	20
MILP-SI	62566	29487	30184	492571	1145508	1771707	1470525	1691075	1694070		4	30
CP-Nod	**90**	**90**	**90**	**33660**	**33622**	**33622**	**511950**	**532192**	**553003**		4	30
MILP-SI	p.	p.	p.	1317658	951745	1052849	1226532	444214	1482423		5	30
CP-Nod	26	26	26	**4885**	**4885**	**4885**	**97399**	**99383**	**101332**		5	30
MILP-SI	p.	p.	p.	2267185	2616084	884031	34210413	46569011	+59652116		6	40
CP-Nod	17	17	17	**1198**	**1198**	**1198**	**35354**	**35354**	**35354**		6	40

Table 5 Simplex iterations performed by the MILP approach (MILP-SI) and Nodes explored by the CP approach (CP-Nod). Bold figures in the tables highlight the approach that produced the best run time. Underlined figures are infeasible instances. +59652116 means that the search has been stopped before infeasibility could be proved, after 30000 secs (8,3 hours)

Fig. 7 Comparison of the average solution time for the CP approach and for the MILP approach as a function of the audit time T

In contrast to what we observed in the first set of 18 instances, where $T = 1$, in this second set of 90 instances, where $T > 1$, the CP approach always dominates the MILP approach in terms of run time. When $T > 1$ the MILP approach does not scale particularly well with respect to N, $\bar{L}$ and ρ. Instances with a large N, T, $\bar{L}$ and ρ require, in fact, up to more than 30000 seconds to be solved. For these instances CP is able to quickly prove optimality or efficiently detect infeasibility. In contrast, CPLEX requires several simplex iterations and a long time to prove infeasibility. The discrepancy between the two approaches for infeasible problems reaches a factor of 3900, that is three orders of magnitude. Although MILP performs better at proving optimality, its performances are still far from those achieved by the CP approach. In fact the discrepancy between the two approaches with respect to feasible problems reaches a factor of 88: almost two orders of magnitude.

5.2 Discussion of results

The results presented indicate that the CP approach is in general more tractable than the mathematical programming one for this class of scheduling problems. The average solution time over all the instances considered is 950 seconds for the MILP approach and 24 seconds for the CP approach. This shows that, on average, CP is about one order of magnitude faster than MILP for the test bed analyzed. A comparison of solution times for the test problems reveals that, as the value of T increases (Fig. 7), CP is orders of magnitude faster than MILP, irrespectively of N, $\bar{L}$ and ρ. It should be noted that CP, as a consequence of constraint propagation, is extremely good at proving infeasibility, while this is the class of problems for which MILP requires significant computational efforts. CP also shows a more stable behavior and scalable performances as T, N, $\bar{L}$ and ρ increase. MILP performs poorly for the largest instances considered both in proving optimality and detecting infeasibility.

The performance of CP-based and MILP-based approaches for solving a number of combinatorial optimization problems has been the scope of many recent studies (e.g., the modified generalized assignment problem [11], the template design problem [24], the progressive party problem [28]). There has been effort to characterise the properties of different problems by their effect on the performance of CP and MILP approaches[11,19]. The key result of that work is that MILP is very efficient when the relaxation is tight and the models have a structure that can be effectively exploited, while CP seems to work better for highly constrained discrete optimization problems in which the expressiveness of MILP is a major limitation. Our results confirm that the best model of choice depends on the characteristics of the instances rather than of the structure exposed at the problem level. Our experiments suggest that when the audit time is small the relaxation is tight, hence MILP performs well; when the audit time gets longer the problem becomes more constrained, and CP seems to scale up much better than MILP.

6 Conclusions

This paper addresses the stochastic combinatorial optimization problem of scheduling internal audit activities. In [30] a related problem has been tackled by means of a similar MILP approach, but the authors assume that audit activities are instantaneous (conducting an audit does not take any time). Our work is more general and more realistic since we consider non-instantaneous audit activities, and we schedule the audit team in such a way as to prevent concurrent auditing.

We proposed a stochastic programming formulation and we developed two alternative certainty equivalent approaches to solve this model: an MILP model and a CP model. Our computational experience shows that MILP proved to be effective when the time required to perform an audit is short ($T \leq 1$). In contrast, our CP approach proved to be very effective when the audit time T is greater than one period. The CP approach proved extremely effective both in proving optimality and detecting infeasibility for most of the instances considered. For instances where the audit time T is greater than one, the CP approach proves optimality or detect infeasibility in a time that is typically orders-of-magnitude less than the one required by the MILP approach. Nevertheless the performance of the CP approach when the audit time is short still remains acceptable and close to that achieved by the MILP approach.

Finally, we believe that introducing additional complexity in the model may constitute an interesting direction for future research. For instance, heterogeneous audit teams may be considered, which may take different times to perform audits; alternatively, random audit durations — rather than a fixed and deterministic duration T — may be incorporated in the stochastic programming model.

7 Appendix

In this section a complete list of the notation adopted in the paper is given.

N : (constant) number of time periods in the planning horizon;

i, j, t : (index) a time period;

M : (constant) number of audit units;

m : (index) an audit unit;

l_t^m : (random variable) a normally distributed random variable representing losses that accrue in audit unit m during period t;

$g_{l_t^m}(l_t^m)$: (function) probability density function of l_t^m;

$E[.]$: (function) expected value operator;

μ_t^m : (constant) expected value of l_t^m, sometimes expressed as $E[l_t^m]$;

σ_t^m : (constant) standard deviation of l_t^m;

ρ : (constant) coefficient of variation of l_t^m, $\rho = \sigma_t^m/\mu_t^m$;

T : (constant) number of time periods required by the audit team to complete an audit;

a : (constant) the fixed cost incurred each time an audit is conducted;

h : (constant) the loss discount factor measuring the opportunity cost associated with a given loss level;

$\bar{L}$: (constant) a threshold indicating the maximum allowed loss level in each period;

α : (constant) the probability of not exceeding the loss threshold $\bar{L}$;

L_t^m : (decision variable) the loss level in audit unit m at the beginning of period t;

$E[TC]$: (objective function) the sum of expected audit costs and discounted period losses that are expected to accrue in the absence of auditing;

K_t^m : (decision variable) a variable that takes the value of 1 if an internal audit (lasting T periods) is started for audit unit m in period t, otherwise 0;

$\mathfrak{M}$: (constant) some very large number;

A_r^m : (index) time period in which the rth audit is performed on unit m;

$G_{l_t^m}(x)$: (function) $G_{l_t^m}(x) = \int_{-\infty}^{x} g_{l_t^m}(\tau)d\tau$ is the cumulative distribution function of l_t^m;

$G_{l_t^m}^{-1}(\alpha)$: (function) the inverse cumulative distribution function (or α-quantile) of l_t^m;

$G_N(.)$: (function) the standard normal distribution function;

$G_N^{-1}(.)$: (function) the inverse of $G_N(.)$;

$P_{t,j}^m$: (decision variable) a binary variable that takes a value of 1 if the most recent audit prior to period t was in period j and zero elsewhere for a given audit unit m;

C : (constant) maximum number of audit teams that the firm can assign to conduct audits in any given time period;

$\Phi[m, t, j]$: (constant table) a 3-dimensional table whose elements are
defined as $\Phi[m, t, j] = \bar{L} - G^{-1}_{l^m_j + l^m_{j+1} + \ldots + l^m_t}(\alpha)$,
$t = 1, \ldots, N, \quad j = 1, \ldots, N;$

References

1. K. Apt. *Principles of Constraint Programming*. Cambridge University Press, New York, NY, USA, 2003.
2. J. R. Birge and F. Louveaux. *Introduction to Stochastic Programming*. Springer Verlag, New York, 1997.
3. J. H. Bookbinder and J. Y. Tan. Strategies for the probabilistic lot-sizing problem with service-level constraints. *Management Science*, 34:1096–1108, 1988.
4. J. E. Boritz and D. S. Broca. Scheduling internal audit activities. *Auditing: A Journal of Practice & Theory*, 6:1–19, 1986.
5. S. C. Brailsford, C. N. Potts, and B. M. Smith. Constraint satisfaction problems: Algorithms and applications. *European Journal of Operational Research*, 119:557–581, 1999.
6. A. Charnes and W. W. Cooper. Chance-constrainted programming. *Management Science*, 6(1):73–79, 1959.
7. Vaek Chvtal. *Linear Programming*. W. H. Freeman & Co., New York, 1983.
8. Alain Colmerauer. Prolog in 10 figures. *Commun. ACM*, 28(12):1296–1310, 1985.
9. G. B. Dantzig. Maximization of a linear function of variables subject to linear inequalities. In *Activity Analysis of Production and Allocation (Chap. XXI)*. Wiley, New York, 1951.
10. G. B. Dantzig. *Linear Programming and Extensions*. Princeton University Press, Princeton, New Jersey, 1963.
11. Ken Darby-Dowman, James Little, Gautam Mitra, and Marco Zaffalon. Constraint logic programming and integer programming approaches and their collaboration in solving an assignment scheduling problem. *Constraints*, 1(3):245–264, 1997.
12. Michael Hanus. Programming with constraints: An introduction by kim marriott and peter j. stuckey, mit press, 1998. *J. Funct. Program.*, 11(2):253–262, 2001.
13. P. Van Hentenryck and J. P. Carillon. Generality vs. specificity: an experience with ai and or techniques. In *Proceedings of the National Conference on Artificial Intelligence (AAAI-88)*, 1988.
14. Pascal Van Hentenryck. *Constraint satisfaction in logic programming*. MIT Press, Cambridge, MA, USA, 1989.
15. Pascal Van Hentenryck. *The OPL optimization programming language*. MIT Press, Cambridge, MA, USA, 1999.
16. J. Hughes. Optimal internal audit timing. *The Accounting Review*, 52:56–68, 1977.
17. Inc. ILOG. *CPLEX 9.0 Users Manual*. ILOG, Inc., Incline Village, NV, 2007.
18. Inc. ILOG. *OPL Studio 3.7 Users Manual*. ILOG, Inc., Incline Village, NV, 2007.

19. V. Jain and I. E. Grossmann. Algorithms for hybrid milp/cp models for a class of optimization problems. *INFORMS Journal on computing*, 13:258–276, 2001.

20. H. Jeffreys. *Theory of Probability*. Clarendon Press, Oxford, UK, 1961.

21. W. R. Knechel and H. P. Benson. The optimization approach for scheduling internal audits of division. *Decision Sciences*, 22:391–405, 1991.

22. I. J. Lustig and J. F. Puget. Program does not equal program: Constraint programming and its relationship to mathematical programming. *Interfaces*, 31:29–53, 2001.

23. G. L. Nemhauser and L. A. Wolsey. *Integer and combinatorial optimization*. Wiley-Interscience, New York, NY, USA, 1988.

24. L. Proll and B. Smith. Integer linear programming and constraint logic programming approaches to a template design problem. 10:265–275, 1998.

25. J.-C. Regin. A filtering algorithm for constraints of difference in csps. In *Proceedings of the National Conference on Artificial Intelligence (AAAI-94)*, pages 362–367, 1994. Seattle, WA, USA.

26. J.-C Regin. *Global Constraints and Filtering Algorithms*. in Constraints and Integer Programming Combined, Kluwer, M. Milano editor, 2003.

27. A. Schrijver. *Theory of Linear and Integer Programming*. John Wiley, New York, 1986.

28. Barbara Smith, Sally Brailsford, Peter Hubbard, and H. Paul Williams. The Progressive Party Problem: Integer Linear Programming and Constraint Programming Compared. In *CP95: Proceedings 1st International Conference on Principles and Practice of Constraint Programming)*, Marseilles, 1995.

29. S. A. Tarim and B. Smith. Constraint Programming for Computing Non-Stationary (R,S) Inventory Policies. *European Journal of Operational Research*. Forthcoming.

30. S. A. Tarim, O. Tayfur, and S. Karacaer. A mip model for scheduling multi-location internal audit activities. Technical report, Hacettepe University, Dept of Management, 2008.

31. E. P. K. Tsang. *Foundations of Constraint Satisfaction*. Academic Press, London and San Diego, 1993.

32. E. S. Ventsel. *Theory of Probability*. Moscow, Nauka, (In Russian), 1979.

33. H. P. Williams. *Model Building in Mathematical Programming (Third Edition)*. John Wiley & Sons, New York, 1994.

34. D. Wilson and R. Ranson. Internal audit scheduling — a mathematical model. *The Internal Auditor*, July-August:42–50, 1971.