# MILP feasibility by nonlinear programming

Leo Liberti[1]

[1] *LIX, École Polytechnique, F-91128 Palaiseau, France*
Email:liberti@lix.polytechnique.fr

December 3, 2017

**Abstract**

We discuss a tightly feasible mixed-integer linear programs arising in the energy industry, for which branch-and-bound appears to be ineffective. We consider its hardness, measure the probability that randomly generated instances are feasible or almost feasible, and introduce heuristic solution methods based on relaxing different constraints of the problem. We show the computational efficiency of our simplest heuristic (a multi-start based on a linear complementarity programming reformulation of the given problem) with respect to a state-of-the-art branch-and-bound implementation.

## 1 Introduction

In this paper we consider the following Mixed-Integer Linear Program (MILP):

$$\left.\begin{array}{rrcl} \min & \sum\limits_{h\le p} y_h & & \\ & Qy & = & x \\ & Ax & \le & b \\ & x^L \le & x & \le x^U \\ & y & \in & \{0,1\}^p, \end{array}\right\} \quad (1)$$

where $Q = (q_{jh})$ is an $n \times p$ real matrix, $A = (a_{ij})$ is an $m \times n$ real matrix, $b \in \mathbb{R}^m$, $x$ is a vector of $n$ continuous decision variables, and $y$ is a vector of $p$ binary decision variables.

It turns out that Eq. (1) is a good model for hydro-energy unit commitment problems with hydro-electrical generating units along a valley [18]. The binary variables $y$ model some on/off controls along a discretized time line. The controls influence (through the equations $Qy = x$) some physical quantities $x$ that are constrained to lie in $[x^L, x^U]$. The decision maker seeks the smallest number of controls that need to be switched on in order for the physical constraints on $x$ to be feasible. According to [18], however, it is numerically really difficult to satisfy the constraints of Eq. (1), at least in the instances arising at Electricité de France (EDF). In particular, the authors detail the efforts of solving Eq. (1) with common MILP solution techniques, such as the Branch-and-Bound (BB) solver CPLEX [11], which would normally be considered the best solution method for such problems.

At first sight, the MILP in Eq. (1) may not strike the reader as a particularly "nasty" problem, insofar as structure goes. The infeasibility issues arise because the instances solved at EDF enforce very tight bounds $[x^L, x^U]$ on $x$ — sometimes requiring that $x^L = x^U$ (which occurs in run-of-the-river reservoirs). Note that the constraints $Ax \le b$ in Eq. (1) are supposed to encode "the rest of the problem" (which is quite extensive, and may involve more decision variables than just the $x$'s). In a private communication [6], we were told that the infeasibilities were mostly related to the problem in Eq. (2), obtained as a relaxation of the original problem in Eq. (1) by shedding the technical constraints. Eq. (2) will be our problem of interest in the rest of this paper.

$$\left.\begin{array}{rrcl} \min & \sum\limits_{h\le p} y_h & & \\ & Qy & = & x \\ & x^L \le & x & \le x^U \\ & y & \in & \{0,1\}^p. \end{array}\right\} \quad (2)$$

Although in this paper we focus exclusively on feasibility, the objective function is discussed in [20]. The choice of limiting our attention to Eq. (2) is also due to computational considerations: the EDF instances for the original problem are confidential, and were not made available to us. We therefore work with instances generated randomly from Eq. (2).

In this paper, we leverage an observation made in [8] about Linear Complementarity Programming (LCP) reformulations of tightly constrained MILPs in binary variables: they often (heuristically) leading to an exact, or almost exact, solution. We discuss the computational and empirical hardness of Eq. (2) and present some solution methodologies. We focus on a specific one based on an LCP reformulation, which we also test computationally.

The rest of this paper is organized as follows. In Sect. 2 we discuss the computational complexity of our problem. In Sect. 3 we consider the difficulty of the problem in terms of the probability of a random instance being feasible in function of the bounds $[x^L, x^U]$. In Sect. 4 we discuss some unusual solution methods that are not based on BB.

## 2 Hardness

Just how hard is the problem in Eq. (2)? We consider the following set of linear equations in binary variables $y \in \{0,1\}^p$:

$$\forall j \le n \qquad \sum_{h\le p} q_{ih} y_h = \bar{x}_j, \quad (3)$$

where $\bar{x} = x^L = x^U$. We assume $Q, \bar{x}$ are rational. We stress that fixing the variables $x$ to a fixed constant $\bar{x}$ appears to be an important practical case [6] in the motivating application. We remark that Eq. (3) is a well-known case of difficult Binary Linear Program (BLP), i.e. the so-called *market split* [5, 1] (a.k.a. "market share").

First, we consider the case $n = 1$, i.e. (3) consists of just one equation. Then Eq. (3) is a rational equivalent of the SUBSET SUM problem [7, §SP13], with instance given by $(Q, \bar{x})$. The fact that $Q, \bar{x}$ are rational obviously does not change the problem: it suffices to rescale all data by the minimum common multiple of all the denominators. This problem is weakly **NP**-complete: it can be solved in pseudopolynomial time by a dynamic programming algorithm [7]. The Wikipedia entry for SUBSET SUM also states that the difficulty of solving this problem depends on the number of variables $p$ and the number of bits necessary to encode $Q$: if either is fixed, the problem becomes tractable. It is well known that Integer Linear Programs (ILP) with a fixed number of variables $p$ can be solved in polynomial time [16].

In this paper, we are interested in the case where $p$ is not fixed, whereas $n$ might be fixed or not. The case with $n$ fixed is relevant for the motivating application, as the operational points $x_j$ on which the constraints $x_j \in [x_j^L, x_j^U]$ are verified could be given by the (fixed) physical properties of the considered hydro valley. We are not aware of results in ILP complexity with a fixed number of constraints but non-fixed number of variables, however.

If neither $n$ nor $p$ is fixed, we note that there is a natural reduction from SAT to a version of Eq. (3) where $Q$ has entries in $\{-1, 0, 1\}$, which shows that solving Eq. (3) is strongly **NP**-complete (by inclusion with respect to $Q$). Again by inclusion (with respect to $x^L, x^U$), Eq. (2) is also **NP**-complete.

Eq. (2) is one of those cases when complexity proofs by inclusion are not quite satisfactory. The empirical hardness of solving Eq. (2) obviously decreases as the bounds $[x^L, x^U]$ grow farther apart (if $x^L = -\infty$ and $x^U = \infty$ any solution of Eq. (2) is trivially feasible). A more convincing complexity proof should take the width parameter $W = \max_{j \le n}(x_j^U - x_j^L)$ into account, too. In Sect. 3 below we attempt to provide a more appropriate hardness measure in terms of the probability of achieving (near-)feasibility in a randomly generated instance.

# 3 Likelihood of approximate feasibility

In this section we consider the probability that uniformly sampled instances of Eq. (3) and Eq. (2) are (almost) feasible.

## 3.1 The Irwin-Hall distribution

Consider Eq. (3). For each $j \le n$ and $h \le p$ we assume that $q_{jh}$ is sampled from a random variable $\tilde{Q}_{jh}$ uniformly distributed in $[0, 1]$. We let

$$\hat{Q}_j = \sum_{\substack{h \le p \\ y_h = 1}} \tilde{Q}_{jh}.$$

We want to derive an expression, which depends on $x^L$ and $x^U$, of the probability that a uniformly sampled instance of Eq. (2) is feasible. To do that, we first look at the case $\bar{x} = x^L = x^U$, consider near-feasibility, and tackle instances where the cardinality of the support of $y$ is fixed to a given integer $K$, i.e. $\sum_{h \le p} y_h = K$. For all $j \le n$, the corresponding random variable is

$$\hat{Q}_j^K = \sum_{\substack{h \le p \\ \sum_h y_h = K}} \tilde{Q}_{jh},$$

i.e. the sum of $K$ i.i.d. uniform random variables on $[0, 1]$.

For any given $j \le n$, the distribution of $\hat{Q}_j^K$ is known as the *Irwin-Hall* distribution [10, 21]. Its mean is $K/2$ and its variance is $K/12$. It can also be shown that for $K > 1$ the probability distribution function (PDF) of $\hat{Q}_j^K$ attains a strict (local) maximum at the mean $K/2$. The cumulative distribution function (CDF) is

$$F_K(x) = \frac{1}{K!} \sum_{k=0}^{\lfloor x \rfloor} (-1)^k \binom{K}{k} (x - k)^K. \tag{4}$$

By Eq. (4), for any $j \le n$ the probability of $\hat{Q}_j^K$ taking values between $x_j^L$ and $x_j^U$ is $F_K(x_j^U) - F_K(x_j^L)$, a quantity we shall denote by $\gamma_K(j)$.

## 3.2 Near-feasibility for $n = 1$

If we fix $n = 1$, understanding the distribution of $\hat{Q}_1^K$ would allow us to glance at some uniformly generated input data, $(Q, \bar{x})$, and get an idea of the likelihood of a given binary vector $y$ with $K$ nonzeroes yielding values close to $\bar{x}_1$.

We first want to make a qualitative statement to the effect that instances where $[x_1^L, x_1^U]$ contains the mean $K/2$ are likely to be easier than those that do not.

**3.1 Lemma**
*There exists a value $r < K$ such that, from $r$ units from the mean, the tails of the probability density function (PDF) of $\hat{Q}^K$ converge to zero exponentially fast.*

*Proof.* First, we argue about the right tail. Trivially, since $\tilde{Q}_{1h}$ is uniformly sampled in $[0,1]$ for each $h \le p$, we have $\mathbb{P}(\hat{Q}_1^K \ge K) = 0$. We now have to argue the negative exponential convergence in some sub-range of $[K/2, K]$ including the right end-point $K$. By [15, Thm. 7.5], we have

$$\mathbb{P}\left(\hat{Q}^K \ge \frac{K}{2} + r\right) \le e^{-\frac{K}{2}g(\frac{2r}{K})} \tag{5}$$

for any $r > 0$, where $g(u) = (1+u)\ln(1+u) - u$ for $u \ge 0$. The exponent in the RHS of Eq. (5) is negative as long as:

$$\left(1 + \frac{2r}{K}\right)\ln\left(1 + \frac{2r}{K}\right) > \frac{2r}{K}.$$

Trivially, we have that, for all $v > e$, $v \ln v > v$. So, if $1 + \frac{2r}{K} > e$, it follows that $(1+2r/K)\ln(1+2r/K) > 1 + 2r/K$, which is obviously strictly greater than $2r/K$. We therefore need $r > K(e-1)/2$ for the statement to hold, as claimed.

The argument for the left tail follows by symmetry of the PDF, which can established by considering the distribution of the sum of $K$ uniform random variables over $[-1, 0]$. □ □

By Lemma 3.1, the measure of the PDF of $\hat{Q}_1^K$ is "concentrated" in $[\frac{K-e}{2}, \frac{K+e}{2}]$. This makes it reasonable to expect that instances will be easier as $\bar{x}$ moves towards $\frac{K}{2}$.

Quantitative statements about the probability of near-feasibility can be obtained for given values of $K$ and $x_1^L, x_1^U$ by evaluating $\gamma_K(1)$.

## 3.3 Feasibility in the general case

We now move back to the general case with $n > 1$ and bounds $x^L, x^U \in \mathbb{R}^n$ on $x$ as in Eq. (2). Recall that for all $j \le n$ we defined

$$\mathbb{P}(\exists y \in \{0,1\}^p \ \hat{Q}_j^K \in [x_j^L, x_j^U] \mid \mathbf{1}y = K) = \gamma_K(j). \tag{6}$$

**3.2 Proposition**
*The probability that a uniformly sampled instance of Eq.* (2) *is feasible is:*

$$\mathbb{P}\big(\exists y \in \{0,1\}^p \text{ s.t. } \tilde{Q}y \in [x^L, x^U]\big) = \frac{1}{2^p}\sum_{K \le p}\binom{p}{K}\Big(1 - \prod_{j \le n}(1 - \gamma_K(j))\Big). \tag{7}$$

*Proof.* The probability that there exists $y$ with support cardinality $K$ that satisfies *all* the constraints is

$$\begin{aligned}
& \mathbb{P}(\exists y \in \{0,1\}^p \ \hat{Q}^K \in [x^L, x^U] \mid \mathbf{1}y = K) = \\
= 1 \quad - \quad & \mathbb{P}(\forall y \in \{0,1\}^p \ \hat{Q}^K \notin [x^L, x^U] \mid \mathbf{1}y = K) = \\
= 1 \quad - \quad & \prod_{j \le n}(1 - \mathbb{P}(\nexists y \in \{0,1\}^p \ \hat{Q}_j^K \in [x_j^L, x_j^U] \mid \mathbf{1}y = K)) = \\
= 1 \quad - \quad & \prod_{j \le n}(1 - \gamma_K(j)),
\end{aligned}$$

where the last equality follows by Eq. (6). We can take the union over all possible values of $K \in \{0, \dots p\}$ by weighing each probability by the probability that a uniformly sampled binary vector $y$ should have support cardinality $K$, i.e. $\binom{p}{K}/2^p$, which yields Eq. (7), as claimed. □ □

For example, if $x^L = 0$ and $x^U = p$, then obviously $\gamma_K(j) = 1$ for each $K \le p$ and $j \le n$, which yields $\frac{1}{2^p}\sum\limits_{K \le p}\binom{p}{k} = 1$, as expected.

# 4 Solution methods

Since Eq. (2) is a MILP, the solution method of choice is the Branch-and-Bound, implemented for example using a state-of-the-art solver such as CPLEX [11]. As mentioned in Sect. 1, however, given the difficulty in finding feasible solutions, CPLEX can rarely prune by bound, which means that it shows the brunt of its exponential behaviour. In this section we present three alternative heuristic methods based on relaxing different constraints of Eq. (2).

## 4.1 Relaxation of integrality constraints

Every BLP

$$\min\{c^\top y \mid By \le \beta \wedge y \in \{0,1\}^p\} \tag{8}$$

can be exactly reformulated to an LCP by replacing the integrality constraints $y \in \{0,1\}^p$ by the following:

$$0 \le y \le 1 \tag{9}$$

$$z = 1 - y \tag{10}$$

$$\sum_{h\le p} y_h z_h = 0. \tag{11}$$

By Eq. (9)-(10) we have $0 \le z \le 1$, which implies that every product in Eq. (11) is non-negative, which in turn means that the sum is non-negative. Thus, the sum is zero if and only if every term is zero, so either $y_h = 0$ or $z_h = 1 - y_h = 0$, i.e. $y_h \in \{0,1\}$ for each $h \le p$. An equivalent Nonlinear Program (NLP) removes the $z$ variables altogether:

$$\min\Big\{c^\top y \mid By \le \beta \wedge \sum_{h\le p} y_h(1-y_h) = 0\Big\}. \tag{12}$$

This provides the following empirically efficient heuristic algorithm for our problem $P$ in Eq. (2):

1. consider the continuous relaxation $\bar{P}$ of $P$;
2. solve $\bar{P}$ in polynomial time, e.g. by the interior point method [12];
3. use the solution $x', y'$ as a starting point for a local NLP solver (e.g. [13]) on the problem

$$\min\Big\{\sum_{h\le p} y_h \mid \text{Eq. (3)} \wedge \sum_{h\le p} y_h(1-y_h) = 0\Big\}. \tag{13}$$

We remark that only Step 3 is crucial: randomly sampling the starting point is also a valid choice, as evidenced by the reasonable success of Multi-Start (MS) methods for global optimization [17].

In practice, we employ a slack variable $s \ge 0$ on the linear complementarity constraint Eq. (11) in Eq. (13):

$$\left.\begin{array}{rrcl}
\min & \sum\limits_{h\le p} y_h + \eta s & & \\
\forall j \le n & \sum\limits_{h\le p} q_{jh} y_h & = & x \\
& \sum\limits_{h\le p} y_h(y_h-1) & = & s \\
& x & \in & [x^L, x^U] \\
& y & \in & \{0,1\}^p \\
& s & \ge & 0,
\end{array}\right\} \tag{14}$$

where $\eta > 0$ is a scaling coefficient chosen empirically. The interest of Eq. (14) is that it always provides a solution: even when $s > 0$ (and hence Eq. (11) is not satisfied), we can always hope that the (fractional) $y$

variables will be close enough to integrality that a rounding will yield a feasible or near-feasible solution. As evidenced in Sect. 5, we consider the solution $(x^*, y^*)$ where $y^*$ is obtained by rounding the $y$ variables, and $x^*$ are re-computed as $Qy^*$ according to Eq. (3).

### 4.1.1 The case of fixed $n$

When $n$ is fixed, we can exploit Barvinok's polynomial-time algorithm for systems with a fixed number of homogeneous quadratic equations [3]. We consider the case where $\bar{x} = x^L = x^U$ (Eq. (3)), together with the constraint

$$\sum_h y_h(y_h - 1) = 0,$$

which we homogenize by adding a new variable $z$ and noting that $y_h^2 = y_h$ for each $h \leq p$ (since $y \in \{0,1\}^p$):

$$\begin{aligned} \forall j \leq n \quad \sum_{h \leq p} q_{ih} y_h^2 &= \bar{x}_{jz}^2 && (15) \\ \sum_{h \leq p} y_h^2 &= \sum_{h \leq p} y_h z. && (16) \end{aligned}$$

We remark that all variables $y, z$ are now continuous and unconstrained, but if we achieve a feasible solution with $y \geq 0$ then Eq. (16) will necessarily imply $y \in \{0,1\}$ and $z = 1$. We deal with the objective function $\min \sum_h y_h$ by replacing each $y_h$ with $y_h^2$, to obtain

$$\min \Big\{ \sum_{h \leq p} y_h^2 \mid \text{Eq. (15)-(16)} \Big\},$$

which we can solve by a bisection method on the objective function value using Barvinok's algorithm as a feasibility oracle. This requires the homogenization of the equation $\sum_h y_h^2 = c$, where $c$ is a constant varied by the bisection method. The system passed to Barvinok's algorithm is therefore

$$\sum_{h \leq p} y_h^2 = cz^2 \quad \wedge \quad \text{Eq. (15)-(16)}.$$

After each call to Barvinok's algorithm, the condition $y \geq 0$ must be verified. If it does not hold, then the heuristic stops inconclusively. Otherwise it identifies the optimum of Eq. (2) up to a given $\epsilon > 0$ in time $O(\log(p/\epsilon)p^n)$, polynomial when $n$ is fixed.

We chose not to implement this heuristic, for three reasons: (i) Barvinok's algorithm seems rather difficult to implement; (ii) it might not be very efficient in case $n$ is fixed but not extremely small; (iii) the derived heuristic does not appear to provide any useful information in case of failure.

## 4.2 Relaxing the $[0,1]$ bounds

We again assume $\bar{x} = x^L = x^U$ as in Eq. (3). We also assume that $Q$ and $\bar{x}$ have integer components, and relax the $y$ variables to take any integer value rather than just binary.

This setting opens up algorithmic opportunities from the field of linear Diophantine equations. There exist appropriately sized unimodular matrices $L, R$ such that $LQR$ is a diagonal matrix $D$ where each nonzero diagonal entry divides the next nonzero diagonal entry [14, Thm. 1]. The system $Qy = \bar{x}$ (with $y \in \mathbb{Z}^p$) can therefore be decomposed into $Dz = L\bar{x}$ and $y = Rz$. Now, assuming one can find $L, R$, solving $Dz = L\bar{x}$ is easy since $D$ is diagonal, and then $y$ can be computed from $Rz$. There are algorithms for computing $L, R$ that work in a cubic number of steps in function of $n, p$ [4].

If we step back to the specificities of the real application, $Q$ consists of floating point numbers given to a precision of at least $10^{-6}$: rescaling would likely yield large integer coefficients, which would probably make the application of solution algorithms for linear Diophantine equations unwieldy. On the other hand, a systematic treatment of classical results about linear Diophantine equations with rational coefficients and unbounded variables is given in [19, Ch. 4]. Imposing bounds on the variables, however, makes the problem hard again [2]. We therefore decided not to implement this method.

# 5 Computational results

Our results aim at ascertaining whether the MS heuristic:

1. sample a starting point $x' \in \mathbb{R}^n$, $y' \in \{0,1\}^p$;
2. deploy a local NLP solver from $x', y'$ on Eq. (14);
3. round the binary solution $y^* = \lfloor y' \rceil$ and compute $x^* = Qy^*$.

is competitive in order to find feasible solutions for Eq. (2), compared with a BB-based solver. To this goal, we randomly generated two sets of instances $I_1, I_2$ of Eq. (2), depending on a vector $\bar{x} \in \mathbb{R}^n$, over the following parameters:

- $n \in \{1, 2, 5, 10, 50, 99\}$;
- $p \in \{10, 50, 100, 500, 999\}$;
- data in $Q$ uniformly sampled from either $[0,1]$ or $[-1,1]$;
- $x^L = \bar{x} - \theta$ and $x^U = \bar{x} + \theta$ for $\theta \in \{0, 0.05, 0.1\}$ and $\bar{x}$ chosen as detailed below.

The first set $I_1$ contains instances that are feasible by construction. This is achieved by defining $\bar{x}$ as follows:

1. sample $y \in \{0,1\}^p$ from $p$ Bernoulli distributions;
2. for all $j \leq n$ compute $\bar{x}_j = \sum_{h \leq p} Q_{jh} y_h$.

The second set $I_2$ contains instances that are infeasible with some probability: this is achieved by sampling each component of $\bar{x}$ from a uniform distribution.

We then solved the instances in $I_1, I_2$ by running the BB solver CPLEX 12.6.3 [11] and the MS heuristic above have on a 4-CPU Intel Xeon X3220 at 2.4GHz with 8GB RAM running Linux. We chose SNOPT 7.2 as local NLP solver [9] in the MS heuristic. We gave CPLEX the default parameters but set the time limit to 180s of "wall-clock" time (we recall that CPLEX is a parallel solver by default, so the actual CPU time measured in the experiments is not limited by 180s but by the actual user CPU time spent, as reported by the operating system).

The performance of BB solvers on infeasible instances is severely impaired by a lack of feasible solution because no node is ever pruned by bound; the overall effect is more similar to a complete enumeration than to the typically "smart" enumeration carried out by such solvers. To avoid penalizing CPLEX for this reason, we employed a reformulation of Eq. (2) which shifts all the infeasibility into slack variables:

$$\left.\begin{array}{rrcl} & \min\limits_{x,y,s} \sum\limits_{h\le p} y_h & + & \sum\limits_{j\le n} (s_j^+ + s_j^-) \\ \forall j \le n & \sum\limits_{h\le p} q_{ih} y_h & = & x_j + s_j^+ - s_j^- \\ & x^L \le & x & \le x^U \\ & s^+, & s^- & \ge 0 \\ & y & \in & \{0,1\}^p, \end{array}\right\} \quad (17)$$

and carried out the same modification on Eq. (14).

The only possible configuration for the MS heuristic is the maximum allowed number $T$ of iterations, which we set to $p$ (the same as the number of binary variables) in order to have the effort depend on the size of the instance.

The results for the feasible instance set $I_1$ are presented in Table 1. We report the instance details ($n$, $p$, the distribution type $\mathsf{distr}$, the $[x^L, x^U]$ range half-width $\theta$), whether each method found a feasible solution ($\mathsf{feas} \in \{0,1\}$), the sum of the infeasibilities w.r.t. Eq. (3) $\mathsf{conerr}$, computed as

$$\mathsf{conerr} = \sum_{j\le n} \big( \max(0, x_j^* - x_j^U) + \max(0, x_j^L - x_j^*) \big),$$

and the CPU times ($\mathsf{CPU}$). An instance is classifed as feasible ($\mathsf{feas} = 1$) if $\mathsf{conerr} < 10^{-8}$ (with also, obviously, $y^* \in \{0,1\}^p$).

We remark that we report 6 decimal digits in Table 1 but the computations have been carried out at the machine floating point precision. We report sum, averages and standard deviations for $\mathsf{feas}$, $\mathsf{conerr}$, $\mathsf{CPU}$ in Table 2. For lack of space we only report sum, average and standard deviations (rather than complete results) for the results on the infeasible instance set $I_2$. It is clear from Tables 2-3 that continuous optimization techniques are extremely beneficial in finding feasible solutions to tightly constrained MILPs.

One might question whether the superior performance of the MS heuristic is due to the higher CPU time effort of MS w.r.t. CPLEX. To ascertain this, we re-ran the experiments with the CPU time of the MS heuristic capped at 3 minutes. Table 4 reports on the sums, averages and variances of results obtained on both $I_1$ and $I_2$ this way. We remark that the results on $I_1$ in Table 4 are actually better than those in Table 2. This is just due to the stochastic nature of the MS heuristic, and to the fact that good optima are identified early on in the search. Overall, we conclude that the CPU time taken by MS is not the reason for its advantage w.r.t. BB.

## 6 Conclusion

We explored the old idea of replacing binary variables by continuous ones bounded by $[0,1]$ and a linear complementarity constraint. For tightly constrained MILPs occurring in the hydro unit commitment problem (similar to market share instances), we show that a simple multi-start approach is superior to a state-of-the-art branch-and-bound solver.

## References

[1] K. Aardal, R. Bixby, C. Hurkens, A. Lenstra, and J. Smeltink. Market split and basis reduction: Towards a solution of the Cornuéjols-Dawande instances. In G. Cornuéjols, R. Burkard, and G. Woeginger, editors, *Integer Programming and Combinatorial Optimization*, volume 1610 of *LNCS*, pages 1–16, Berlin, 1999. Springer.

[2] K. Aardal, C. Hurkens, and A. Lenstra. Solving a system of linear Diophantine equations with lower and upper bounds on the variables. *Mathematics of Operations Research*, 25(3):427–442, 2000.

[3] A. Barvinok. Feasibility testing for systems of real quadratic equations. *Discrete and Computational Geometry*, 10:1–13, 1993.

[4] G. Bradley. Algorithms for Hermite and Smith normal matrices and linear Diophantine equations. *Mathematics of Computation*, 25(116):897–907, 1971.

[5] G. Cornuéjols and M. Dawande. A class of hard small 0-1 programs. In R. Bixby, E. Boyd, and R. Ríos-Mercado, editors, *Integer Programming and Combinatorial Optimization*, volume 1412 of *LNCS*, pages 284–293, Berlin, 1998. Springer.

[6] C. D'Ambrosio. Personal communication, 2017.

[7] M. Garey and D. Johnson. *Computers and Intractability: a Guide to the Theory of* **NP**-*Completeness*. Freeman and Company, New York, 1979.

[8] L. Di Giacomo, G. Patrizi, and E. Argento. Linear complementarity as a general solution method to combinatorial problems. *INFORMS Journal on Computing*, 19(1):73–79, 2007.

[9] P.E. Gill. *User's guide for SNOPT version 7.2*. Systems Optimization Laboratory, Stanford University, California, 2006.

[10] P. Hall. The distribution of means for samples of size $n$ drawn from a population in which the variate takes values between 0 and 1, all such values being equally probable. *Biometrika*, 19(3/4):240–245, 1927.

[11] IBM. *ILOG CPLEX 12.6 User's Manual*. IBM, 2014.

[12] B. Jansen, C. Roos, T. Terlaky, and J.-Ph. Vial. Interior-point methodology for linear programming: duality, sensitivity analysis and computational aspects. Technical Report 28, TU Delft, 1993.

[13] M. Kojima, N. Megiddo, and Y. Ye. An interior point potential reduction algorithm for the linear complementarity problem. *Mathematical Programming*, 54:267–279, 1992.

[14] F. Lazebnik. On systems of linear Diophantine equations. *Mathematics Magazine*, 69(4):261–266, 1996.

[15] M. Ledoux. *The concentration of measure phenomenon*. Number 89 in Mathematical Surveys and Monographs. American Mathematical Society, Providence, 2005.

[16] H. Lenstra. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8(4):538–548, 1983.

[17] M. Locatelli and F. Schoen. Random linkage: a family of acceptance/rejection algorithms for global optimization. *Mathematical Programming*, 85(2):379–396, 1999.

[18] Y. Sahraoui, P. Bendotti, and C. D'Ambrosio. Real-world hydro-power unit-commitment: dealing with numerical errors and feasibility issues. *Energy*, accepted.

[19] A. Schrijver. *Theory of Linear and Integer Programming*. Wiley, Chichester, 1986.

[20] R. Taktak and C. D'Ambrosio. On overview on mathematical programming approaches for the deterministic unit commitment problem in hydro valleys. *Energy Systems*, 8(1):57–79, 2017.

[21] Wikipedia. Irwin-hall distribution, 2017. [Online; accessed 171119].

| | | | | feas | | conerr | | CPU | |
|---|---|---|---|---|---|---|---|---|---|
| $n$ | $p$ | distr | $\theta$ | BB | MS | BB | MS | BB | MS |
| 1 | 10 | 0 | 0 | 0 | 0 | 2.32407 | 0.061345 | 0.01 | 0.04 |
| 1 | 10 | 0 | 0.05 | 0 | 0 | 1.578351 | 0.161354 | 0.01 | 0.04 |
| 1 | 10 | 0 | 0.1 | 0 | 1 | 2.700129 | 0 | 0.01 | 0.04 |
| 1 | 10 | 1 | 0 | 0 | 0 | 1.261148 | 0.064085 | 0.01 | 0.04 |
| 1 | 10 | 1 | 0.05 | 0 | 0 | 1.211148 | 0.018378 | 0.01 | 0.04 |
| 1 | 10 | 1 | 0.1 | 0 | 1 | 1.161148 | 0 | 0.01 | 0.03 |
| 1 | 50 | 0 | 0 | 0 | 0 | 11.349767 | 0.002907 | 0.02 | 0.24 |
| 1 | 50 | 0 | 0.05 | 0 | 1 | 13.913338 | 0 | 0.07 | 0.19 |
| 1 | 50 | 0 | 0.1 | 0 | 1 | 13.863338 | 0 | 0.1 | 0.16 |
| 1 | 50 | 1 | 0 | 0 | 0 | 2.790113 | 2.383729 | 0.02 | 0.24 |
| 1 | 50 | 1 | 0.05 | 0 | 0 | 2.740113 | 2.740113 | 0.03 | 0.24 |
| 1 | 50 | 1 | 0.1 | 0 | 1 | 1.16734 | 0 | 0.01 | 0.26 |
| 1 | 100 | 0 | 0 | 0 | 0 | 28.007851 | 0.016543 | 0.02 | 0.54 |
| 1 | 100 | 0 | 0.05 | 0 | 0 | 30.865685 | 0.048186 | 0.02 | 0.41 |
| 1 | 100 | 0 | 0.1 | 0 | 0 | 19.610965 | 0.0284 | 0.02 | 0.66 |
| 1 | 100 | 1 | 0 | 0 | 0 | 8.57807 | 0.131922 | 0.02 | 0.63 |
| 1 | 100 | 1 | 0.05 | 0 | 0 | 3.120869 | 3.076816 | 0.02 | 0.63 |
| 1 | 100 | 1 | 0.1 | 0 | 1 | 3.070869 | 0 | 0.02 | 0.62 |
| 1 | 500 | 0 | 0 | 0 | 0 | 123.493014 | 0.00468 | 0.04 | 14.86 |
| 1 | 500 | 0 | 0.05 | 0 | 1 | 123.443014 | 0 | 0.03 | 16.78 |
| 1 | 500 | 0 | 0.1 | 0 | 1 | 119.406109 | 0 | 0.05 | 36.97 |
| 1 | 500 | 1 | 0 | 0 | 0 | 12.720343 | 2.896958 | 0.02 | 31.37 |
| 1 | 500 | 1 | 0.05 | 0 | 1 | 7.066126 | 0 | 0.02 | 30.91 |
| 1 | 500 | 1 | 0.1 | 0 | 1 | 7.016126 | 0 | 0.02 | 31 |
| 1 | 999 | 0 | 0 | 0 | 0 | 247.036395 | 0.046231 | 0.04 | 100.45 |
| 1 | 999 | 0 | 0.05 | 0 | 1 | 255.617392 | 0 | 0.04 | 21.53 |
| 1 | 999 | 0 | 0.1 | 0 | 1 | 234.886437 | 0 | 0.04 | 175.02 |
| 1 | 999 | 1 | 0 | 0 | 0 | 0.713901 | 0.006678 | 0.03 | 213.72 |
| 1 | 999 | 1 | 0.05 | 0 | 0 | 19.365855 | 0.033537 | 0.03 | 247.51 |
| 1 | 999 | 1 | 0.1 | 0 | 1 | 7.904365 | 0 | 0.03 | 229.83 |
| 2 | 10 | 0 | 0 | 0 | 0 | 0.351547 | 0.359571 | 0.03 | 0.04 |
| 2 | 10 | 0 | 0.05 | 0 | 0 | 0.618234 | 0.254019 | 0.04 | 0.03 |
| 2 | 10 | 0 | 0.1 | 0 | 0 | 0.518234 | 0.254314 | 0.15 | 0.03 |
| 2 | 10 | 1 | 0 | 0 | 0 | 1.081735 | 0.204043 | 0.02 | 0.03 |
| 2 | 10 | 1 | 0.05 | 0 | 0 | 0.303347 | 0.137929 | 0.1 | 0.03 |
| 2 | 10 | 1 | 0.1 | 0 | 0 | 0.203347 | 0.163569 | 0.14 | 0.05 |
| 2 | 50 | 0 | 0 | 0 | 0 | 1.484605 | 0.05694 | 0.04 | 0.13 |
| 2 | 50 | 0 | 0.05 | 0 | 0 | 0.391349 | 0.040117 | 0.04 | 0.14 |
| 2 | 50 | 0 | 0.1 | 0 | 0 | 0.072481 | 0.146071 | 0.16 | 0.16 |
| 2 | 50 | 1 | 0 | 0 | 0 | 0.693182 | 0.202292 | 0.02 | 0.19 |
| 2 | 50 | 1 | 0.05 | 0 | 0 | 0.593182 | 0.0489 | 0.03 | 0.2 |
| 2 | 50 | 1 | 0.1 | 0 | 0 | 0.493182 | 0.065863 | 0.02 | 0.2 |
| 2 | 100 | 0 | 0 | 0 | 0 | 0.323233 | 0.345594 | 0.04 | 0.33 |
| 2 | 100 | 0 | 0.05 | 0 | 0 | 0.682022 | 0.012533 | 0.24 | 0.48 |
| 2 | 100 | 0 | 0.1 | 0 | 1 | 0.582022 | 0 | 0.09 | 0.47 |
| 2 | 100 | 1 | 0 | 0 | 0 | 5.287322 | 0.130741 | 0.26 | 0.5 |
| 2 | 100 | 1 | 0.05 | 0 | 1 | 5.187322 | 0 | 0.2 | 0.51 |
| 2 | 100 | 1 | 0.1 | 0 | 0 | 0.289258 | 0.088442 | 0.16 | 0.49 |
| 2 | 500 | 0 | 0 | 0 | 0 | 0.752074 | 0.155689 | 0.73 | 4.84 |
| 2 | 500 | 0 | 0.05 | 0 | 0 | 0.042095 | 0.071255 | 0.07 | 25.27 |
| 2 | 500 | 0 | 0.1 | 1 | 0 | 0 | 0.022983 | 0.24 | 11.27 |
| 2 | 500 | 1 | 0 | 0 | 0 | 5.937512 | 0.099361 | 0.13 | 33.01 |
| 2 | 500 | 1 | 0.05 | 0 | 0 | 5.858057 | 0.08288 | 0.17 | 33.2 |
| 2 | 500 | 1 | 0.1 | 0 | 0 | 4.42073 | 0.037246 | 0.11 | 33.57 |
| 2 | 999 | 0 | 0 | 0 | 0 | 5.9E-05 | 0.088107 | 410.11 | 184.17 |
| 2 | 999 | 0 | 0.05 | 0 | 1 | 0.935847 | 0 | 406.56 | 27.81 |
| 2 | 999 | 0 | 0.1 | 0 | 1 | 0.835847 | 0 | 410.47 | 27.44 |
| 2 | 999 | 1 | 0 | 0 | 0 | 0.000609 | 0.065714 | 2.09 | 189.23 |
| 2 | 999 | 1 | 0.05 | 0 | 0 | 13.992941 | 0.01872 | 0.13 | 220.97 |
| 2 | 999 | 1 | 0.1 | 0 | 1 | 4.289912 | 0 | 0.28 | 251.13 |
| 5 | 10 | 0 | 0 | 0 | 0 | 0.869037 | 0.706022 | 0.19 | 0.03 |
| 5 | 10 | 0 | 0.05 | 0 | 0 | 0.650297 | 0.680799 | 0.18 | 0.03 |
| 5 | 10 | 0 | 0.1 | 0 | 0 | 0.921514 | 0.148645 | 0.12 | 0.03 |
| 5 | 10 | 1 | 0 | 0 | 0 | 2.468394 | 1.775934 | 0.02 | 0.03 |
| 5 | 10 | 1 | 0.05 | 0 | 0 | 2.218394 | 1.525934 | 0.02 | 0.03 |
| 5 | 10 | 1 | 0.1 | 0 | 0 | 1.998247 | 1.251902 | 0.02 | 0.03 |
| 5 | 50 | 0 | 0 | 0 | 0 | 0.287349 | 0.960493 | 0.5 | 0.19 |
| 5 | 50 | 0 | 0.05 | 0 | 0 | 0.755586 | 0.432848 | 3.23 | 0.22 |
| 5 | 50 | 0 | 0.1 | 0 | 0 | 0.538463 | 0.148599 | 0.72 | 0.21 |
| 5 | 50 | 1 | 0 | 0 | 0 | 1.183309 | 0.377512 | 0.21 | 0.22 |
| 5 | 50 | 1 | 0.05 | 0 | 0 | 1.155268 | 0.347119 | 0.29 | 0.22 |
| 5 | 50 | 1 | 0.1 | 0 | 0 | 1.005268 | 0.415876 | 0.28 | 0.22 |
| 5 | 100 | 0 | 0 | 0 | 0 | 0.051365 | 0.445371 | 19 | 0.47 |
| 5 | 100 | 0 | 0.05 | 1 | 0 | 0 | 0.246848 | 71.8 | 0.5 |
| 5 | 100 | 0 | 0.1 | 0 | 0 | 0.951273 | 0.171545 | 149.01 | 0.48 |
| 5 | 100 | 1 | 0 | 0 | 0 | 1.198647 | 0.440855 | 0.31 | 0.6 |
| 5 | 100 | 1 | 0.05 | 0 | 0 | 1.036505 | 0.414626 | 0.54 | 0.64 |
| 5 | 100 | 1 | 0.1 | 0 | 0 | 0.61102 | 0.118224 | 0.16 | 0.64 |
| 5 | 500 | 0 | 0 | 0 | 0 | 0.057685 | 0.284417 | 436.47 | 7.96 |
| 5 | 500 | 0 | 0.05 | 0 | 0 | 0.881686 | 0.092001 | 427.44 | 16.16 |
| 5 | 500 | 0 | 0.1 | 1 | 1 | 0 | 0 | 427.28 | 8.45 |
| 5 | 500 | 1 | 0 | 0 | 0 | 0.58195 | 0.378982 | 2.79 | 32.87 |
| 5 | 500 | 1 | 0.05 | 0 | 0 | 0.085931 | 0.160789 | 2.37 | 36.13 |
| 5 | 500 | 1 | 0.1 | 0 | 0 | 0.928858 | 0.001169 | 1.28 | 33.43 |
| 5 | 999 | 0 | 0 | 0 | 0 | 0.033254 | 0.154364 | 430.61 | 36.43 |
| 5 | 999 | 0 | 0.05 | 1 | 0 | 0 | 0.050479 | 431.69 | 78.5 |
| 5 | 999 | 0 | 0.1 | 1 | 1 | 0 | 0 | 447.98 | 48.52 |
| 5 | 999 | 1 | 0 | 0 | 0 | 3.022855 | 0.103025 | 23.28 | 182.44 |
| 5 | 999 | 1 | 0.05 | 1 | 0 | 0 | 0.022399 | 422.48 | 172.09 |
| 5 | 999 | 1 | 0.1 | 0 | 0 | 0.445252 | 0.058864 | 5.24 | 167.99 |

| | | | | feas | | conerr | | CPU | |
|---|---|---|---|---|---|---|---|---|---|
| $n$ | $p$ | distr | $\theta$ | BB | MS | BB | MS | BB | MS |
| 10 | 10 | 0 | 0 | 1 | 1 | 0 | 0 | 0.03 | 0.03 |
| 10 | 10 | 0 | 0.05 | 1 | 1 | 0 | 0 | 0.02 | 0.03 |
| 10 | 10 | 0 | 0.1 | 1 | 1 | 0 | 0 | 0.03 | 0.04 |
| 10 | 10 | 1 | 0 | 1 | 1 | 0 | 0 | 0.01 | 0.03 |
| 10 | 10 | 1 | 0.05 | 0 | 1 | 2.99655 | 0 | 0.12 | 0.04 |
| 10 | 10 | 1 | 0.1 | 0 | 1 | 2.49655 | 0 | 0.12 | 0.04 |
| 10 | 50 | 0 | 0 | 0 | 0 | 0.519431 | 1.520239 | 3.78 | 0.25 |
| 10 | 50 | 0 | 0.05 | 0 | 0 | 0.172402 | 1.375297 | 2.34 | 0.29 |
| 10 | 50 | 0 | 0.1 | 0 | 0 | 0.049552 | 0.545895 | 12.69 | 0.28 |
| 10 | 50 | 1 | 0 | 0 | 0 | 4.284423 | 2.995239 | 0.5 | 0.25 |
| 10 | 50 | 1 | 0.05 | 0 | 0 | 3.830222 | 1.804413 | 0.52 | 0.27 |
| 10 | 50 | 1 | 0.1 | 0 | 0 | 3.418451 | 1.151005 | 0.13 | 0.29 |
| 10 | 100 | 0 | 0 | 0 | 0 | 1.165771 | 1.655809 | 424.62 | 0.63 |
| 10 | 100 | 0 | 0.05 | 0 | 0 | 0.051527 | 0.590638 | 437.08 | 0.71 |
| 10 | 100 | 0 | 0.1 | 0 | 0 | 0.004725 | 0.7542 | 263.62 | 0.75 |
| 10 | 100 | 1 | 0 | 0 | 0 | 1.599585 | 1.867458 | 0.88 | 0.73 |
| 10 | 100 | 1 | 0.05 | 0 | 0 | 1.914359 | 1.71108 | 1.32 | 0.77 |
| 10 | 100 | 1 | 0.1 | 0 | 0 | 2.127105 | 1.178457 | 1.28 | 0.83 |
| 10 | 500 | 0 | 0 | 0 | 0 | 0.44991 | 1.007725 | 434.48 | 14.73 |
| 10 | 500 | 0 | 0.05 | 0 | 0 | 1.170809 | 0.356041 | 435.92 | 14.47 |
| 10 | 500 | 0 | 0.1 | 1 | 0 | 0 | 0.23927 | 444.33 | 20.1 |
| 10 | 500 | 1 | 0 | 0 | 0 | 2.521863 | 1.317721 | 446.46 | 34.07 |
| 10 | 500 | 1 | 0.05 | 0 | 0 | 0.225351 | 1.231162 | 453.87 | 39.32 |
| 10 | 500 | 1 | 0.1 | 0 | 0 | 0.120738 | 1.100889 | 450.9 | 35.55 |
| 10 | 999 | 0 | 0 | 0 | 0 | 0.652774 | 0.812867 | 458.13 | 55.9 |
| 10 | 999 | 0 | 0.05 | 0 | 0 | 0.113543 | 0.305648 | 463.93 | 64.47 |
| 10 | 999 | 0 | 0.1 | 0 | 0 | 0.288677 | 0.212521 | 453.56 | 71.53 |
| 10 | 999 | 1 | 0 | 0 | 0 | 0.787394 | 1.312658 | 465.24 | 221.48 |
| 10 | 999 | 1 | 0.05 | 0 | 0 | 0.353932 | 1.065403 | 466.37 | 231.37 |
| 10 | 999 | 1 | 0.1 | 0 | 0 | 0.248057 | 0.537679 | 460.1 | 247.08 |
| 50 | 10 | 0 | 0 | 1 | 1 | 0 | 0 | 0.02 | 0.07 |
| 50 | 10 | 0 | 0.05 | 1 | 1 | 0 | 0 | 0.04 | 0.09 |
| 50 | 10 | 0 | 0.1 | 1 | 1 | 0 | 0 | 0.04 | 0.09 |
| 50 | 10 | 1 | 0 | 1 | 1 | 0 | 0 | 0.02 | 0.06 |
| 50 | 10 | 1 | 0.05 | 1 | 1 | 0 | 0 | 0.04 | 0.08 |
| 50 | 10 | 1 | 0.1 | 1 | 1 | 0 | 0 | 0.04 | 0.09 |
| 50 | 50 | 0 | 0 | 1 | 1 | 0 | 0 | 0.03 | 1.23 |
| 50 | 50 | 0 | 0.05 | 1 | 1 | 0 | 0 | 0.17 | 2.3 |
| 50 | 50 | 0 | 0.1 | 1 | 1 | 0 | 0 | 0.3 | 2.2 |
| 50 | 50 | 1 | 0 | 1 | 1 | 0 | 0 | 0.03 | 1.08 |
| 50 | 50 | 1 | 0.05 | 1 | 1 | 0 | 0 | 0.1 | 1.76 |
| 50 | 50 | 1 | 0.1 | 1 | 1 | 0 | 0 | 0.28 | 1.85 |
| 50 | 100 | 0 | 0 | 1 | 0 | 0 | 15.823588 | 0.93 | 3.69 |
| 50 | 100 | 0 | 0.05 | 1 | 1 | 0 | 0 | 8.24 | 5.89 |
| 50 | 100 | 0 | 0.1 | 1 | 0 | 0 | 10.970566 | 2.88 | 6.02 |
| 50 | 100 | 1 | 0 | 1 | 1 | 0 | 0 | 1.23 | 3.36 |
| 50 | 100 | 1 | 0.05 | 1 | 1 | 0 | 0 | 0.77 | 4.2 |
| 50 | 100 | 1 | 0.1 | 1 | 0 | 0 | 26.752762 | 10.21 | 4.23 |
| 50 | 500 | 0 | 0 | 0 | 0 | 13.027666 | 12.909458 | 460.91 | 96.71 |
| 50 | 500 | 0 | 0.05 | 0 | 0 | 11.216089 | 9.970944 | 477.07 | 154.97 |
| 50 | 500 | 0 | 0.1 | 0 | 0 | 9.325591 | 7.851837 | 461.81 | 148.07 |
| 50 | 500 | 1 | 0 | 0 | 0 | 24.342004 | 26.182309 | 476.6 | 130.92 |
| 50 | 500 | 1 | 0.05 | 0 | 0 | 27.068901 | 22.683359 | 483.7 | 189.23 |
| 50 | 500 | 1 | 0.1 | 0 | 0 | 20.61643 | 21.493069 | 490.75 | 199.8 |
| 50 | 999 | 0 | 0 | 0 | 0 | 13.108549 | 11.284734 | 495.09 | 411.8 |
| 50 | 999 | 0 | 0.05 | 0 | 0 | 11.684617 | 9.066446 | 483.66 | 690.39 |
| 50 | 999 | 0 | 0.1 | 0 | 0 | 9.552165 | 7.052495 | 480.33 | 641.75 |
| 50 | 999 | 1 | 0 | 0 | 0 | 27.06203 | 22.467814 | 491.73 | 838.73 |
| 50 | 999 | 1 | 0.05 | 0 | 0 | 24.044542 | 20.977076 | 496.45 | 1122.32 |
| 50 | 999 | 1 | 0.1 | 0 | 0 | 21.310585 | 18.977143 | 494.81 | 1221.51 |
| 99 | 10 | 0 | 0 | 1 | 1 | 0 | 0 | 0.02 | 0.11 |
| 99 | 10 | 0 | 0.05 | 1 | 1 | 0 | 0 | 0.05 | 0.18 |
| 99 | 10 | 0 | 0.1 | 1 | 1 | 0 | 0 | 0.05 | 0.17 |
| 99 | 10 | 1 | 0 | 1 | 1 | 0 | 0 | 0.03 | 0.11 |
| 99 | 10 | 1 | 0.05 | 1 | 1 | 0 | 0 | 0.04 | 0.19 |
| 99 | 10 | 1 | 0.1 | 1 | 1 | 0 | 0 | 0.07 | 0.19 |
| 99 | 50 | 0 | 0 | 1 | 1 | 0 | 0 | 0.04 | 1.96 |
| 99 | 50 | 0 | 0.05 | 1 | 1 | 0 | 0 | 0.25 | 6.15 |
| 99 | 50 | 0 | 0.1 | 1 | 1 | 0 | 0 | 0.21 | 6.29 |
| 99 | 50 | 1 | 0 | 1 | 1 | 0 | 0 | 0.04 | 1.71 |
| 99 | 50 | 1 | 0.05 | 1 | 1 | 0 | 0 | 0.21 | 5.68 |
| 99 | 50 | 1 | 0.1 | 1 | 1 | 0 | 0 | 0.5 | 5.86 |
| 99 | 100 | 0 | 0 | 1 | 1 | 0 | 0 | 0.44 | 13.19 |
| 99 | 100 | 0 | 0.05 | 1 | 1 | 0 | 0 | 1.82 | 34.09 |
| 99 | 100 | 0 | 0.1 | 1 | 1 | 0 | 0 | 1.8 | 34.3 |
| 99 | 100 | 1 | 0 | 1 | 1 | 0 | 0 | 0.36 | 8.94 |
| 99 | 100 | 1 | 0.05 | 1 | 1 | 0 | 0 | 0.7 | 27.18 |
| 99 | 100 | 1 | 0.1 | 1 | 1 | 0 | 0 | 1.89 | 26.98 |
| 99 | 500 | 0 | 0 | 0 | 0 | 35.075665 | 36.372957 | 524.1 | 347.02 |
| 99 | 500 | 0 | 0.05 | 0 | 0 | 35.640828 | 34.034148 | 520.66 | 681.15 |
| 99 | 500 | 0 | 0.1 | 0 | 0 | 30.475635 | 28.59999 | 522.43 | 708.54 |
| 99 | 500 | 1 | 0 | 0 | 0 | 81.70416 | 80.200879 | 525.74 | 433.17 |
| 99 | 500 | 1 | 0.05 | 0 | 0 | 67.853744 | 76.015859 | 524.03 | 658.92 |
| 99 | 500 | 1 | 0.1 | 0 | 0 | 72.285066 | 72.912895 | 525.64 | 718.14 |
| 99 | 999 | 0 | 0 | 0 | 0 | 38.724098 | 37.083856 | 523.66 | 1712.5 |
| 99 | 999 | 0 | 0.05 | 0 | 0 | 34.098302 | 31.210423 | 524.42 | 3002.4 |
| 99 | 999 | 0 | 0.1 | 0 | 0 | 30.495419 | 25.813992 | 530.95 | 2990.05 |
| 99 | 999 | 1 | 0 | 0 | 0 | 76.365674 | 78.565427 | 520.15 | 2547.18 |
| 99 | 999 | 1 | 0.05 | 0 | 0 | 67.236768 | 69.93739 | 525.46 | 4078.99 |
| 99 | 999 | 1 | 0.1 | 0 | 0 | 67.9875 | 65.822184 | 538.52 | 4580.88 |

Table 1: Comparative detailed result on the feasible instance set $I_1$ with $T = p$ MS iterations.

| | feas | | conerr | | CPU | |
|---|---|---|---|---|---|---|
| *sum* | 47 | **59** | 2270.01 | **934.47** | **23254** | 32600 |
| *avg* | 0.261 | **0.328** | 12.61 | **5.19** | **129.2** | 181.1 |
| *stdev* | *0.44* | *0.47* | *36.95* | *14.78* | *209.4* | *610.3* |

Table 2: Sums, averages and standard deviations for the instance set $I_1$ (Table 1).

| | feas | | conerr | | CPU | |
|---|---|---|---|---|---|---|
| *sum* | 0 | **4** | 275542.20 | **274858.03** | **211.44** | 5985.9 |
| *avg* | 0 | **0.022** | 1530.79 | **1526.99** | **1.17** | 33.26 |
| *stdev* | *0* | *0.148* | *3827.90* | *3831.38* | *8.93* | *73.71* |

Table 3: Sums, averages and standard deviations for the instance set $I_2$ (recall that these instances are generated infeasible with high probability).

| | Feasible instances $I_1$ | | | Infeasible instances $I_2$ | | |
|---|---|---|---|---|---|---|
| | feas | conerr | CPU | feas | conerr | CPU |
| *sum* | 81 | 916.08 | 8395.9 | 6 | 274834.36 | 3916.4 |
| *avg* | 0.45 | 5.09 | 46.64 | 0.03 | 1526.86 | 21.758 |
| *stdev* | *0.50* | *15.33* | *93.22* | *0.18* | *3831.52* | *37.88* |

Table 4: Sums, averages and standard deviations for MS on $I_1, I_2$ capped at 180s of total (user) time.