Nantes Université

Computing Project

-

First year of Master's Degree in Optimization and Operations Research

LS2N - Laboratory of Digital Sciences of Nantes

Team MODELIS - Modélisation Optimisation et DEcision pour la Logistique, l'Industrie et les Services

---

MO-MILP Primal Heuristics

Improvement of Gravity Machine

Author:
Erwan Meunier[1]

Under the direction of:
Xavier Gandibleux[2]
Saïd Hanafi[3]

Defended on April 20, 2023

[1] erwan.meunier@etu.univ-nantes.fr
[2] xavier.gandibleux@univ-nantes.fr
[3] said.hanafi@uphf.fr

Contents

1 Abstract

2 Introduction

3 Definitions and Notations

We consider the problem as presented in [Gandibleux et al., 2021]. The multi-objective 0-1 linear optimisation problems with p objectives (p-01LP) considered can be formulated as follows:

$$\begin{aligned} \min z(x) &= Cx \\ \text{subject to } Ax &\leqq b \\ x &\in \{0,1\}^n \end{aligned}$$

where

- $x \in \{0,1\}^n$, the vector of n binary variables $x_j, j = 1, \ldots, n$;
- $A \in \mathbb{R}^{m \times n}$, the m constraints $A_i x \leq b_i,\ i = 1, \ldots, m$ and $b \in \mathbb{R}^m$;
- $C \in \mathbb{R}^{p \times n}$, the objective matrix where $p \geq 2$;
- $X := \{x \in \{0,1\}^n \mid Ax \leqq b\} \subseteq \mathbb{R}^n$, the set of feasible solutions, with $\mathbb{R}^n$ the decision space;
- $Y := \{Cx \mid x \in X\} \subseteq \mathbb{R}^p$, the outcome set, with $\mathbb{R}^p$ the objective space.

Thereafter, p is usually meant to be equal to 2 (*e.g.* we consider the bi-objective class of problems).

Definition 1 ([Ehrgott, 2005])**.** *A feasible solution $\hat{x} \in \mathcal{X}$ is called efficient or Pareto optimal if there is no other $x \in \mathcal{X}$ such that $f(x) \leq f(\hat{x})$. If $\hat{x}$ is efficient, $f(\hat{x})$ is called nondominated point. If $x^1, x^2 \in \mathcal{X}$ and $f(x^1) \leq f(x^2)$ we say x^1 dominates x^2 and $f(x^1)$ dominates $f(x^2)$. The set of all efficient solutions $\hat{x} \in \mathcal{X}$ is denoted $\mathcal{X}_E$ and called the efficient set. The set of all nondominated points $\hat{y} = f(\hat{x}) \in \mathcal{Y}$, where $\hat{x} \in \mathcal{X}_E$, is denoted $\mathcal{Y}_N$ and called the nondominated set.*

Definition 2 ([Ehrgott, 2005])**.** *A feasible solution $\hat{x} \in \mathcal{X}$ is called weakly efficient (weakly Pareto optimal) if there is no $x \in \mathcal{X}$ such that $f(x) < f(\hat{x})$, i.e. $f_k(x) < f_k(\hat{x})$ for all $k = 1, \ldots, p$. The point $\hat{y} = f(\hat{x})$ is then called weakly nondominated. A feasible solution $\hat{x} \in \mathcal{X}$ is called strictly efficient (strictly Pareto optimal) if there is no $x \in \mathcal{X}$,$x \neq \hat{x}$ such that $f(x) \leqq f(\hat{x})$. The weakly (strictly) efficient and nondominated sets are denoted $\mathcal{X}_{wE}(\mathcal{X}_{sE})$ and $\mathcal{Y}_{wE}$, respectively.*

Definition 3 (A lower bound set [Ehrgott and Gandibleux, 2001])**.** *A lower bound set for Y' is a subset $L \subseteq \mathbb{R}^p$*

1. *For each $y \in Y'$ there is some $l \in L$ such that $l \leqq y$*
2. *There is no pair $y \in Y', l \in L$ such that y dominates l.*

Definition 4 (An upper bound set [Ehrgott and Gandibleux, 2001])**.** *An upper bound set for Y' is a subset $U \subset \mathbb{R}^p$*

1. *For each $y \in Y'$ there is some $u \in U$ such that $y \leqq u$*
2. *There is no pair $y \in Y', \in U$ such that u dominates y.*

Definition 5. *Let $(\mathcal{X}, f, \mathbb{Z}^p)$ / id / $(\mathbb{Z}^p, \leq)$ be a multiobjective optimization problem of the Pareto class and $\mathcal{X}_E$ be the efficient set and and $\mathcal{Y}_N$ be the nondominated set.*

1. *Let $x \in \mathcal{X}_E$. If there is some $\lambda \in \mathbb{R}^p_>$ such that $x \in \mathcal{X}_E$ is an optimal solution of $\min_{x\in\mathcal{X}} \lambda^T f(x)$ then x is called a* supported efficient solution *and $y = f(x)$ is called* supported nondominated point*. The sets of all supported efficient solutions and supported nondominated points are denoted $\mathcal{X}_{sE}$ and $\mathcal{Y}_{sN}$, respectively. Otherwise, x and y are called* nonsupported*, the notations are $\mathcal{X}_{nE}$ and $\mathcal{Y}_{nN}$*

2. *x^1 and x^2 are called equivalent if $f(x^1) = f(x^2)$. A complete set of efficient solutions is any subset $\mathcal{X}' \subset \mathcal{X}$ such that $f(\mathcal{X}) = \mathcal{Y}_N$. A* minimal complete set *is a complete set without any equivalent solutions. $\mathcal{X}_E$ is also called the* maximal complete set*.*

3. *If x is a supported efficient solution and $y = f(x)$ is an extreme point of $\boldsymbol{conv}(\mathcal{Y})$ then x is called an* extreme supported efficient solution*. y is an* extreme nondominated point*.*

Definition 6 (Ideal and Nadir points)**.**

1. *The point $y^I = \left(y^I_1, \ldots, y^I_p\right)$ given by*

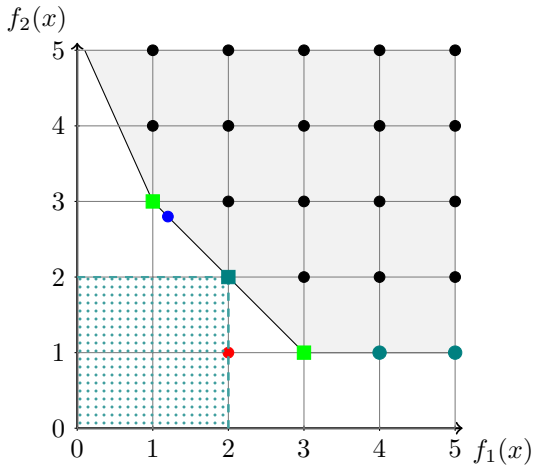
$$y^I_k := \min_{x\in\mathcal{X}} f_k(x) = \min_{y\in\mathcal{Y}} y_k \tag{1}$$

is called the ideal point *of the multicriteria optimization problem $min_{x\in\mathcal{X}} (f_1(x), \ldots, f_2(x))$.*

2. *The point $y^N = \left(y^N_1, \ldots, y^N_p\right)$ given by*

$$y^N_k := \max_{x\in\mathcal{X}_E} f_k(x) = \max_{y\in\mathcal{Y}_N} y_k \tag{2}$$

is called the nadir point *of the multicriteria optimization problem.*



blabla

4 Gravity Machine

Gravity Machine is an algorithm aiming to compute an upperbound set for a multi-objective linear optimization problem with binary variables. It is based the on the *Feasibilty Pump* [Fischetti et al., 2005] which is a heuristic for finding a feasible solution of a given MIP. The essence of this well studied heuristic [Berthold et al., 2019] is given by:

Algorithm 1: Feasibilty Pump (basic version)

Data: termination_criteria, $T \in \mathbb{N}^*$
Result: A feasible solution or nothing

begin
  $nIT \leftarrow 0$;
  $x^* \leftarrow \text{argmin}\left\{c^T x : Ax \geq b\right\}$;
  if x^**is integer* **then**
    return x^*
  else
    $\tilde{x} \leftarrow [x^*]$;
    while $\neg$*termination_criteria* **do**
      $nIT \leftarrow nIT + 1$;
      $x^* \leftarrow \text{argmin}\left\{\Delta(x, \tilde{x}) : Ax \geq b\right\}$;
      if x^**is integer* **then**
        return x^*;
      else if $\exists j \in \mathcal{I} : \left[x_j^*\right] \neq \tilde{x}_j$ **then**
        $\tilde{x} \leftarrow [x^*]$
      else
        $\mathcal{X} \leftarrow \text{SORT}\left(\mathcal{I}, \text{by highest } \left|x_j^* - \tilde{x}_j\right|\right)$;
        $k \leftarrow \text{RAND}\left(\frac{T}{2}, \frac{3T}{2}\right)$;
        FLIP the k first entries $\tilde{x}_j \in \mathcal{X}$;
      end
    end
  end
end

4.1 Generators

4.2 Projection

4.3 Rounding

4.4 Perturbing

5 Improving generators

Generators are obtained by using the ϵ-constraints method which is one of the well-known scalarization techniques [Chankong and Haimes, 1983] aimed at getting a sample of efficient solutions of a MOLP. Those generators form an efficient set for the relaxed problem. That is to say, the set of all generators is also the lower bound set L. Since GM starts the search based on each k-th generator, one would like to have a better initial solution in order to reach a feasible solution faster.

Algorithm 2: Outline of Gravity Machine

Data: $\mathcal{D}$,n_L,maxTrial,maxTime
Result: L,U

begin
  $L, F \leftarrow$ ComputeGenerators$(\mathcal{D}, n_L)$;
  forall $z(\bar{x}^k) \in L, \bar{x}^k \notin F$ **do** //each k-th unfeasible gen
    timeStart $\leftarrow$ Time(); trial $\leftarrow 0$;
    timeout $\leftarrow$ false; feasible $\leftarrow$ false;
    $\tilde{x}, H, cycle \leftarrow$ RoundingSolution$(\bar{x}^k, H)$;
    while $\neg feasible \wedge \neg timeout$ **do**
      trial $\leftarrow$ trial $+1$;
      $\bar{x}, F$,feasible $\leftarrow$ ProjectSolution$(\tilde{x})$;
      if $\neg feasible$ **then**
        $\tilde{x}, H, cycle \leftarrow$ RoundingSolution$(\bar{x}, k, h)$;
        if *cycle* **then**
          $\tilde{x} \leftarrow$ PerturbSolution$(\tilde{x})$;
        end
      end
      timeout $\leftarrow$ (time()-timeStart $\geq$ maxTime) $\vee$ (trial=maxTrial);
    end
  end
  $U \leftarrow$ ExtractNonDominatedPoint(F);
end

5.1 Bugs in Gravity Machine

5.1.1 Generators not feasible

The previous version of Gravity Machine did use the GLPK solver to generate the lower bound set by the ϵ-constraints method. However, closer to the line passing through the Nadir and the origin the more constrained the MILP representing the current generator. Since considered problems have lot of constraints and variables some numerical issues may arise [Gurobi Optimization, LLC, 2023]. Infeasibility hence may appear during the solving process for problems which has at least one (feasible) optimal solution. In addition, the *Julia* wrapper for GLPK seems to be known for returning the solution minimizing constraints violation[4]. Finally, as the termination status was not correctly handled, some generators turn out to be used whereas there were not "trully" feasible.

Patch: The termination status is now considered and GM properly fails if a generator is not feasible. Tolerance attribute was hand-tuned to allow the solver to find a feasible solution. . It is worth noticing that no difference in the performance has been observed. The robustness is the matter here. In addition, GLPK was replaced by HiGHS [Huangfu and Hall, 2018]. Indeed, the latter shows better performances than GLPK [Mittelman, 2023] and is still open-source.

[EM] ça serait bien que ce ne soit pas "hand-tuned" https://discourse.julial both-primal-and-dual-feasibility-tolerance-according-to-the-size-of-the-problem/97455

[4] https://discourse.julialang.org/t/access-infeasible-optimization-results-glpk-with-semicontinuous-variable/59972

5.1.2 Soft conic constraints λ_1 and λ_2

Both vectors λ_1 and λ_2 were accidentaly swaped in their definition such that the "guidance mechanism" was useless and a fortiori misleading.

5.2 Empirical findings

Empirical evidences tend to show that very few variables are set to non-integral value for each generator.

5.3 Setting some variables binary

At first sight, non-solved to integrality variables should be set binary and the problem re-optimized. The latter allows GM to find initial solution Then, a generator closer to Y should result from a the re-optimization process. However, several issues arises:

5.3.1 Communicating vessels effect

Since not all constraints are set binary, some variables found to be zero or one might become fractional. A number of variables found fractional after the second stage solving greater than the number of variables found fractional after the first stage solving represents the worst case. As far as our experimental results show us, this situation is very rare. [EM] à reformuler

5.3.2 Same generators lead to the same solution

Ideally, one would that each generator $\bar{y}^k$ converges to a different $\tilde{y}^{k,t}$ for some iterations $t \in \mathbb{N}$. Below, we consider $J_0(x)$ (and $J_1(x)$) the set containing the index of variables set to zero (respectively to one) into the solution x. Some well known modelization techniques are briefly presented and their pertinence for GM is studied. For the major part of them, they are inspired by canonical cut constraints introduced by [Balas and Jeroslow, 1972] and used for the same purpose by [Hanafi and Wilbaut, 2011].

Canonical cut constraint:

Proposition 1. *Let x^0 be a vector in $\{0,1\}^n$. The following inequality*

$$\sum_{j \in J_1(x^0)} x_j + \sum_{j \in J_1(x^0)} 1 - x_j \geq 1 \tag{3}$$

cutts off solution x^0 without cutting off any other solution in $\{0,1\}^n$.

Proof. Let's suppose that x^0 is not cutted off by the constraint. Then $x = x^0$ implies $0 \geq 1$ from (1). The cutting off is proved.
Finally, a solution $x^1 \neq_1 x^0$ is assumed to be cutted off by the constraint. That is to say

$$\sum_{j \in J_1(x^0)} x_j^1 + \sum_{j \in J_1(x^0)} 1 - x_j^1 = 0 \tag{4}$$

$$\iff \|x^0 - x^1\| = 0 \tag{5}$$

In other words $x^0 = x^1$ and $x^0 \neq x^1$ which is impossible. Then, 1 holds. □

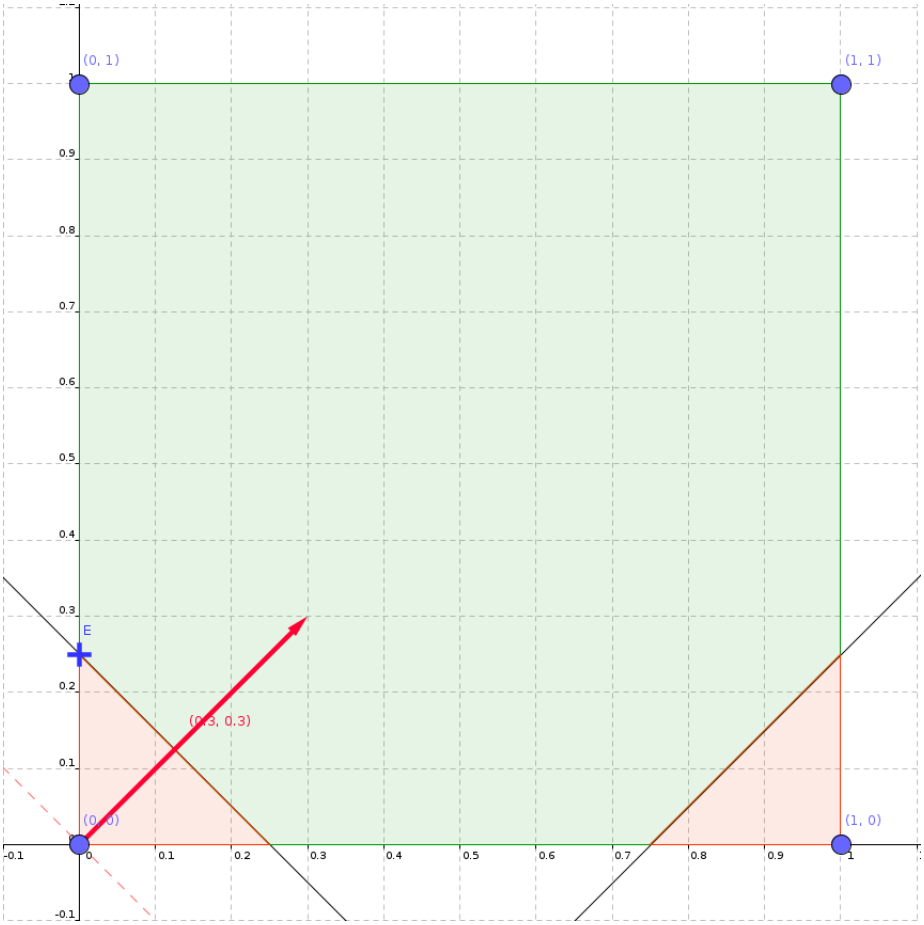


Figure 1: Feasible region

Proposition 2. *Gravity Machine does not longer guarantee the rectitude of L if* $x^0 \in [0,1]^n$

Proof. We exhibit a counter-example. Let be a 1-01LP of the form:

$$\text{Min} \quad x_1 + x_2 \tag{6}$$

$$s.t \quad x_1 + x_2 \geq \frac{1}{4} \tag{7}$$

$$-x_1 + x_2 \geq -\frac{3}{4} \tag{8}$$

$$x_1,\ x_2 \in \{0,1\} \tag{9}$$

According to 5.3.2, $x^* = (0,1)$ is clearly the optimal solution whereas the optimal solution for the relaxed problem is $x^0 = \left(0, \frac{1}{4}\right)$. Since, x_1 has a continuous value, only x_2 is present in the new constraint:

$$1 - x_2 \geq 1 \Leftrightarrow x_2 = 0 \tag{10}$$

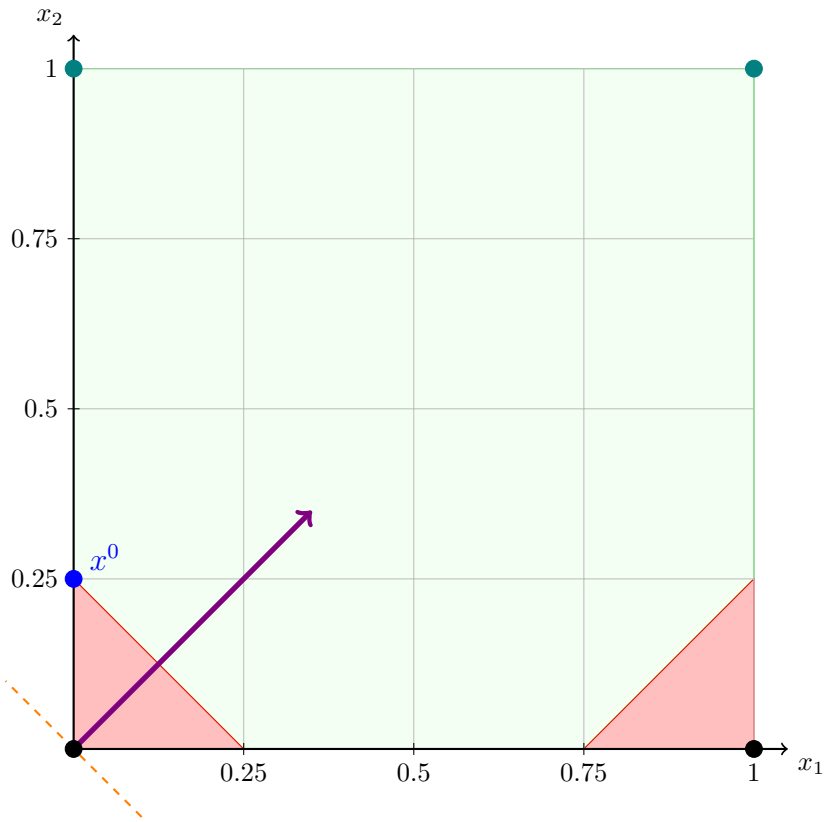


Figure 2:

Consequently, this new constraint prevent the solver finding the x^* and what is more make the problem infeasible. □

Provided a feasible solution s (known to be binary), the following constraint allows the solving process to avoid returning s **only** as a solution [Balas and Jeroslow, 1972]:

$$\sum_{j \in \{i | s_i = 0\}} x_j + \sum_{j \in \{i | s_i = 1\}} 1 - x_j \geq 1$$

It is The lower bound set is no longer guaranteed

Cones

5.3.3 Too much binary variables increase the solving time

As shown by our empirical data, less than 20

Algorithm 3: Improving generators

Data: L,λ_1,λ_2,τ,α
Result: L_I,U
begin
  $L, F \leftarrow$ ComputeGenerators$(\mathcal{D}, n_L)$;
  forall $z(\bar{x}^k) \in L, \bar{x}^k \notin F$ **do** //each k-th unfeasible gen
  end
end

6 Improving projection

By the same token as the generators improvement, the projection used during GM can be improved by adding some integrity constraints on the non

7 Numerical experiments

Gravity Machine is implemented in Julia 1.8.5 [Bezanson et al., 2017]

7.1 Numerical instances

7.2 Quality measure

7.3 Numerical results

Experiments have been performed on-top of the following configuration:

- Laptop: Dell - XPS 13 9300
- OS: Linux Mint 21.1 Cinnamon - v5.6.8
- Kernel version: 5.19.0-38-generic
- Intel© Core™ i7-1065G7 CPU @ 1.30GHz × 4
- L1: 320 KiB, L2: 2 MiB, L3: 8 MiB
- RAM: 15.2 Go LDDR4
- SSD: 1024.2 Go

8 Conclusion and discussion

References

[Balas and Jeroslow, 1972] Balas, E. and Jeroslow, R. (1972). Canonical cuts on the unit hypercube. *SIAM Journal on Applied Mathematics*, 23(1):61–69.

[Berthold et al., 2019] Berthold, T., Lodi, A., and Salvagnin, D. (2019). Ten years of feasibility pump, and counting. *EURO Journal on Computational Optimization*, 7(1):1–14.

[Bezanson et al., 2017] Bezanson, J., Edelman, A., Karpinski, S., and Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *SIAM Review*, 59(1):65–98.

[Chankong and Haimes, 1983] Chankong, V. and Haimes, Y. Y. (1983). Multiobjective decision making: Theory and methodology.

[Ehrgott, 2005] Ehrgott, M. (2005). *Multicriteria Optimization (2. ed.)*. Springer.

[Ehrgott and Gandibleux, 2001] Ehrgott, M. and Gandibleux, X. (2001). Bounds and bound sets for biobjective combinatorial optimization problems. In Köksalan, M. and Zionts, S., editors, *Multiple Criteria Decision Making in the New Millennium*, pages 241–253, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Fischetti et al., 2005] Fischetti, M., Glover, F., and Lodi, A. (2005). The feasibility pump. *Mathematical Programming*, 104(1):91–104.

[Gandibleux et al., 2021] Gandibleux, X., Gasnier, G., and Hanafi, S. (2021). A primal heuristic to compute an upper bound set for multi-objective 0-1 linear optimisation problems. Online. Springer.

[Gurobi Optimization, LLC, 2023] Gurobi Optimization, LLC (2023). Gurobi Guidelines for Numerical Issues - Gurobi Optimization gurobi optimizer reference manual. `https://www.gurobi.com/documentation/9.0/refman/num_grb_guidelines_for_num.html`. [Accessed 06-Apr-2023].

[Hanafi and Wilbaut, 2011] Hanafi, S. and Wilbaut, C. (2011). Improved convergent heuristics for the 0-1 multidimensional knapsack problem. *Annals of Operations Research*, 183(1):125–142.

[Huangfu and Hall, 2018] Huangfu, Q. and Hall, J. A. J. (2018). Parallelizing the dual revised simplex method. *Mathematical Programming Computation*, 10(1):119–142.

[Mittelman, 2023] Mittelman, H. (2023). Benchmarks for optimization software - milp benchmark. `http://plato.asu.edu/ftp/milp.html`. [Accessed 14-Apr-2023].

9 Annexe

Instance	Time	Quality	Number of supported points found
1	416.393	84.42	5
3	249.077	87.3	5
4	Time out	X	X
6	23.874	92.56	5
7	22.625	86.69	4
8	0.452	88.55	3
9	8.225	85.52	5
10	0.647	86.27	2
11	11.838	95.09	3
12	0.208	90.62	5
13	42.496	89.26	5
14	622.631	73.35	4
15	0.319	100	2
16	Time out	X	X
17	599.07	51.51	3
18	46.637	84.56	2
19	3.779	91.72	4
20	1.182	92.68	3
21	0.821	95.45	4
22	1.337	93.49	5
23	0.47	91.43	2
24	2.666	96.13	4
25	2.266	91.12	5
26	0.894	93.41	4
27	1.658	91.15	2
28	1.943	92.4	3
29	6.527	88.21	4
30	6.254	97.06	4
31	13.176	92.66	5
32	0.12	96.99	3
33	14.603	94.62	4
34	2.143	92.29	4
35	4.992	93.43	4
36	5.125	91.59	3
37	1.644	94.88	4
38	2.685	92.43	4
39	1.618	98.16	5
40	0.426	86.44	4
41	0.125	86.77	5
42	1.895	97.69	3
43	6.023	93.31	4

Table 1: Instances solved exactly with 6 solutions at most