Why Unit Two-Variable-Per-Inequality (UTVPI) Constraints Are So Efficient to Handle: Intuitive Explanation

Saeid Tizpaz-Niari, Martine Ceberio, Olga Kosheleva, and Vladik Kreinovich

Abstract In general, integer linear programming is NP-hard. However, there exists a class of integer linear programming problems for which an efficient algorithm is possible: the class of so-called unit two-variable-per-inequality (UTVPI) constraints. In this paper, we provide an intuitive explanation for why an efficient algorithm turned out to be possible for this class. Namely, the smaller the class, the more probable it is that a feasible algorithm is possible for this class, and the UTVPI class is indeed the smallest – in some reasonable sense described in this paper.

1 Formulation of the Problem

Linear programming. In many practical situations, we are interested in checking whether a given system of linear inequalities

$$a_{1,1} \cdot x_1 + \ldots + a_{1,n} \cdot x_n \le b_1,$$

$$\ldots$$

$$a_{m,1} \cdot x_1 + \ldots + a_{m,n} \cdot x_n \le b_m,$$

with given $a_{i,j}$ and b_i, has a solution. For example, linear inequalities correspond to constraints on an engineering design, and we want to check if it is at all possible to find a solution that satisfies all these constraints. Such problems are known as prob-

Saeid Tizpaz-Niari, Martine Ceberio, and Vladik Kreinovich
Department of Computer Science, University of Texas at El Paso, 500 W. University
El Paso, Texas 79968, USA, e-mail: saeid@utep.edu, mceberio@utep.edu, vladik@utep.edu

Olga Kosheleva
Department of Teacher Education, University of Texas at El Paso, 500 W. University
El Paso, Texas 79968, USA, e-mail: olgak@utep.edu

lems of *linear programming*. There exist efficient – polynomial time – algorithms for solving these problems; see, e.g., [5].

Integer linear programming. In the usual formulation of linear programming, variables x_i can take any real values. However, in some practical applications, they can only take integer values. For example, x_i may be the number of workers that need to be added to (or subtracted from) a task. In such cases, usually, it also makes sense to assume that the coefficients $a_{i,j}$ and b_i are integers. Problems under such restriction are known as problems of *integer linear programming*.

It is known that, in general, these problems are NP-hard; see, e.g., [1].

Feasibly solvable classes of integer linear programming problems. There exist classes of integer linear programming problems for which there is a feasible algorithm for checking consistency. One of such classes is the class of Unit Two-Variable-Per-Inequality (UTVPI) constraints; see, e.g., [3, 4]. In this class, each inequality has one of the following forms:

$$x_i \le b, \;\; -x_i \le b, \;\; x_i + x_j \le b, \;\; x_i - x_j \le b, \;\; -x_i - x_j \le b. \tag{1}$$

Interestingly, if we allow slightly more general inequalities of the type

$$a_i \cdot x_i + a_j \cdot x_j \le b,$$

with arbitrary integer a_i, then the problem of checking consistency becomes NP-hard; see, e.g., [2].

Natural question. Technically, proofs are there, but it would be nice to have, in addition to the proofs, an intuitive explanation of these results: why inequalities of type (1) allow a feasible algorithm while other types of inequalities don't?

What we do in this paper. In this paper, we provide a possible intuitive explanation that answers at least some of these questions.

2 Towards an Intuitive Explanation

General idea. As we have mentioned, for the class L of *all* possible linear integer inequalities

$$a_1 \cdot x_1 + \ldots + a_n \cdot x_n \le b, \tag{2}$$

with integer a_i and b (and x_i), the problem of checking whether the system is consistent is NP-hard. We are interesting in finding subclasses of this class for which:

- if we only allow inequalities from this subclass,
- then consistency checking becomes feasible.

In general, if a class of problem allows a feasible algorithm, then this same algorithm works for all subclasses of this class. On the other hand, if we keep increasing the

size of the class, then eventually, we may reach a class for which the corresponding problem is NP-hard. So, sufficiently small classes allow feasible algorithms, while for sufficiently large algorithms, the problem becomes NP-hard. From this viewpoint, the smaller the class, the larger the chances that this class allows a feasible algorithm.

Thus, if we want to find a subclass of the class L for which we a feasible algorithm is possible, our best chance is to look for the smallest subclasses, i.e., for the subclasses with the smallest number of elements.

Two properties of a natural subclass. Which subclasses are natural?

First, let us recall that the ordering of the variables x_i is usually absolutely arbitrary. Because of this, whether an inequality belongs to this subclass or not should not depend on this ordering. In other words, for any permutation $\pi : \{1,\ldots,n\} \mapsto \{1,\ldots,n\}$:

- if the inequality (2) belongs to this subclass,
- then the permuted inequality

$$a_1 \cdot x_{\pi(1)} + \ldots + a_n \cdot x_{\pi(n)} \le b \tag{3}$$

should also belong to this class.

Second, in many occasions, the sign of each variable x_i is chosen arbitrary. For example:

- We can interpret x_i as the number of workers that need to be added to the task. In this case, if we do need to add workers, then x_i is positive, and if we need to move some workers to other tasks, the value x_i should be negative.
- Alternatively, we can interpret x_i as the number of workers that need to be removed from the task. In this case, if we need to add workers, the value x_i becomes negative, but if we need to move some workers to other tasks, the value x_i should be positive.

In precise terms, what was x_i in the first interpretation becomes $-x_i$ in the second one, and vice versa. If we replace x_i with $-x_i'$ in inequality (2), then the sign at the i-th variable changes: $a_i \cdot x_i$ becomes $(-a_i) \cdot x_i'$. Since, as we have mentioned, the sign of each variable is chosen arbitrarily, it makes sense to require that:

- if an inequality (2) belongs to this subclass,
- then, for each i, an inequality obtained from (2) by changing the sign of the i-th variable

$$a_1 \cdot x_1 + \ldots + a_{i-1} \cdot x_{i-1} + (-a_i) \cdot x_i + a_{i+1} \cdot x_{i+1} + \ldots + a_n \cdot x_n \le b \tag{4}$$

should also belong to this class.

So, we arrive at the following definition.

Definition 1. *A non-empty subclass of the class L of all integer inequalities (2) is called* natural *if it is closed under any permutation (3) and under changing the sign of each variable (4).*

Since we are looking for the smallest possible natural subclasses, it makes sense to consider natural subclasses for which no proper subclass is natural.

Definition 2. *We say that a natural subclass S is* basic *if no proper subclass of S is natural.*

Let us describe all possible natural subclasses. The following result describes all possible natural subclasses.

Proposition 1. *Each basic natural subclass S is described by:*

- *a positive integer $k \le n$,*
- *positive integers $k_1, \ldots, k_m$ for which $k_1 + \ldots + k_m = k$, and*
- *m different positive integers $p_1, \ldots, p_m$.*

Elements of this class are obtained by permutation from inequalities of the type

$$\varepsilon_1 \cdot p_1 \cdot x_1 + \ldots + \varepsilon_{k_1} \cdot p_1 \cdot x_{k_1} +$$

$$\varepsilon_{k_1+1} \cdot p_2 \cdot x_{k_1+1} + \ldots + \varepsilon_{k_1+k_2} \cdot p_2 \cdot x_{k_1+k_2} +$$

$$\ldots +$$

$$\varepsilon_{k_1+\ldots+k_{m-1}+1} \cdot p_m \cdot x_{k_1+\ldots+k_{m-1}+1} + \ldots +$$

$$\varepsilon_{k_1+\ldots+k_{m-1}+(k_m-1)} \cdot p_m \cdot x_{k_1+\ldots+k_{m-1}+(k_m-1)} + \varepsilon_k \cdot p_m \cdot x_k \le b, \qquad (5)$$

where $\varepsilon_j \in \{-1, 1\}$.

Proof. Indeed, let S be any basic natural subclass, and let us pick any inequality from this class.

- Let k denote the number of non-zero coefficients a_i in this inequality.
- Let m denote the number of different absolute values of the non-zero coefficients.
- Let $p_1, \ldots, p_m$ denote these different absolute values.
- For each j from 1 to m, let k_j denote the number of variables for which $|a_i| = p_j$.

Then, we can perform the following permutation:

- first, we place all k_1 indices for which $|a_i| = p_1$,
- then, we place all k_2 indices for which $|a_i| = p_2$, etc.,
- finally, we place all k_m indices for which $|a_i| = p_m$,
- and after that, we place all indices (if any) for which $a_i = 0$.

Since the class S is natural, after this permutation, we still get an inequality from the class S. This permuted inequality has exactly the representation (5), with $\varepsilon_i \stackrel{\text{def}}{=} a_i / p_j$, where j is the index for which $|a_i| = p_j$.

Since the original class S was natural, all inequalities obtained from (5) by permutations also belong to the class S. So, the class S' of all these inequalities is a subclass of the class S. One can easily check that this class S' is indeed closed under permutations and under changing signs. So, by Definition 1, the class $S' \subseteq S$ is natural. Since the class S is basic, this means that S' cannot be a proper subclass of the class S. Thus, we have $S' = S$. The proposition is proven.

How many elements are in each basic natural class. Since we are interested in finding basic natural classes with the smallest number of elements, let us find out what is the size of each such class.

Proposition 2. *Each basic natural class has*

$$2^k \cdot \frac{n!}{k_1! \cdot \ldots \cdot k_m! \cdot (n-k)!} \tag{6}$$

elements.

Proof. To count number of elements in the class, let us first count the number of elements in which all the coefficients are positive. For each such element, we can select one of the two signs for each of the k non-zero coefficients, so we get 2^k different inequalities.

To get one of the all-positive elements, we need to enumerate all possible permutations. To describe such a permutation, first, we need to select k_1 out of n elements for which we will have $a_i = p_1$. According to basic combinatorics, the number of such elements is equal to:

$$\frac{n!}{k_1! \cdot (n-k_1)!}. \tag{7}$$

For each such selection, we need to select k_2 out of the not-yet–assigned $n-k_1$ variables. This leads to the following number of choices

$$\frac{(n-k_1)!}{k_2! \cdot (n-k_1-k_2)!}, \tag{8}$$

etc. As a result, we get the following number of permutations:

$$\frac{n!}{k_1! \cdot (n-k_1)!} \cdot \frac{(n-k_1)!}{k_2! \cdot (n-k_1-k_2)!} \cdot \ldots \cdot \frac{(n-k_1-\ldots-k_{m-1})!}{k_m! \cdot (n-k_1-\ldots-k_m)!}. \tag{9}$$

One can see that all the terms $(n-k_1)!$, $(n-k_1-k_2)!$, $\ldots$, $(n-k_1-\ldots-k_{m-1})!$ appear both in the numerator and in the denominator of the product (9) and thus, cancel each other. So, the only remaining terms in the formula (9) are:

$$\frac{n!}{k_1! \cdot \ldots \cdot k_m! \cdot (n-k_1-\ldots-k_m)!}.$$

Since $k_1 + \ldots + k_m = k$, this number is equal to

$$\frac{n!}{k_1! \cdot \ldots \cdot k_m! \cdot (n-k)!}.$$

As we have mentioned, for each such permutation, we have 2^k inequalities corresponding to different combinations of signs, so overall we get exactly the formula (6).

The proposition is proven.

Now, we can find out which class has the smallest number of elements. We need to take into account that since the complexity of the problem increases with n, we are mainly interested in cases when n is sufficiently large.

Proposition 3. *When $n > 5$, in a basic natural class with the smallest number of elements, all inequalities have the form $\varepsilon_i \cdot p_1 \cdot x_i \le b$.*

Proof. For each k, the number of different elements in a basic natural class is described by the formula (6). Similar to our argument in the proof of Proposition 2, the ratio

$$\frac{k!}{k_1! \cdot \ldots k_m!}$$

represents the number of ways we can divide k indices into groups of $k_1, \ldots, k_m$. This number is always greater than or equal to 1, and it is only equal to 1 when we do not have any such division, i.e., when $m = 1$ and $k_1 = k$. Thus, for each k, we have

$$2^k \cdot \frac{n!}{k_1! \cdot \ldots \cdot k_m! \cdot (n-k)!} \ge 2^k \cdot \frac{n!}{k! \cdot (n-k)!},$$

and when $m > 1$, it is a strict inequality. Thus, for each k, the smallest possible size if when $m = 1$. In this case, the number (6) of elements in a class is described by the formula

$$2^k \cdot \frac{n!}{k! \cdot (n-k)!}. \tag{10}$$

It is known that the factor

$$\frac{n!}{k! \cdot (n-k)!}$$

strictly increases for $k \le n/2$ and strictly decreases for $k \ge n/2$. For $k \le n/2$, the factor 2^k also increases. So, the product of these factors also increases, and thus, out of all values $k \le n/2$, the smallest value is attained for $k = 1$, it is equal to $2n$.

For $k \ge n/2$, the smallest value of the second factor is attained when $k = n$, in which case this factor is 1. In this case, the product is equal to 2^n which, for $n > 5$, is always larger that $2n$. Thus, the case $k = n$ cannot correspond to the smallest subclass size. The next smallest value of the second factor is when $k = n - 1$, in this case the second factor is equal to n. In this case, the product is equal to $2^{n-1} \cdot n$ which, for $n > 5$, is also larger than $2n$.

For $k = n - 2$, the second factor is equal to

$$\frac{n \cdot (n-1)}{2}.$$

When $n > 5$, we have

$$\frac{n-1}{2} > 2$$

and thus,

$$\frac{n \cdot (n-1)}{2} > 2n.$$

Thus, this case also cannot be the smallest. For k between $n/2$ and $n-2$, the second factor is even larger, so we also have more elements than in the case of $k=1$. The description of each such class provided in Proposition 1 leads to the conclusion that its inequalities indeed have the desired type. The proposition is proven.

From the viewpoint of checking consistency, this smallest class is trivial. Our objective is to check consistency. From this viewpoint, the above class is not interesting: if each inequality is about only one variable, then checking consistency simply means checking that inequalities corresponding to each variable are consistent. This is trivial, since each inequality of the type $a \cdot x_i \le b$ is equivalent:

- to $x_i \le \lfloor b/p_1 \rfloor$ is $\varepsilon_i = 1$, and
- to $\lfloor b/p_1 \rfloor \le x_i$ if $\varepsilon_i = -1$.

Let us look for the smallest non-trivial class. It therefore makes sense to looks for the smallest non-trivial class.

Definition 3. *We say that a class of inequalities is* non-trivial *if at least one of its inequalities contains at least two variables with non-zero coefficients.*

Proposition 4. *For sufficiently large n, in a non-trivial basic natural class with the smallest number of elements, all inequalities have the form*

$$\varepsilon_i \cdot p_1 \cdot x_i + \varepsilon_j \cdot p_1 \cdot x_j \le b. \tag{11}$$

Proof. Non-trivial means that we cannot have $k=1$, so we must have $k/ge2$. As in the proof of Proposition 3, we can conclude that the smallest number of elements is when $m=1$. In this case, as in the previous proof, for $k \le n/2$, the smallest number corresponds to $k=2$, when we have

$$2^2 \cdot \frac{n \cdot (n-1)}{2} = 2 \cdot n \cdot (n-1)$$

elements. For $k \ge n/2$, the first factor is at least $2^{n/2}$, and for large n, this is larger than $2 \cdot n \cdot (n-1)$, since exponential function grows faster than a polynomial. Thus, the smallest number of elements is indeed attained when $k=2$. The use of Proposition 1 completes the proof.

This leads to the desired explanation. Each inequality (11) is equivalent to

$$\varepsilon_i \cdot x_i + \varepsilon_j \cdot x_j \le \lfloor b/p_1 \rfloor,$$

i.e., to two-variable inequalities of type (1). Thus, the class of inequalities (1) – for which a feasible consistency-checking algorithm is possible – can be easily explained. Namely, in (1), we consider:

- inequalities from the trivial basic natural classes, and
- inequalities form the smallest non-trivial basic natural classes.

Comment. Of course, our discussions do not provide a proof that a feasible algorithm is possible for inequalities of type (1), it just provides the following explanation of why a feasible algorithm turned out to exist for this particular class:

- it is reasonable to expect that the smaller the class, the more probable it is that a feasible algorithm is possible for this class, and
- the class (1) is indeed the smallest – in some reasonable sense described in this paper.

Acknowledgments

This work was supported in part by the National Science Foundation grants 1623190 (A Model of Change for Preparing a New Generation for Professional Practice in Computer Science), HRD-1834620 and HRD-2034030 (CAHSI Includes), EAR-2225395, and by the AT&T Fellowship in Information Technology.

It was also supported by the program of the development of the Scientific-Educational Mathematical Center of Volga Federal District No. 075-02-2020-1478, and by a grant from the Hungarian National Research, Development and Innovation Office (NRDI).

References

1. Th. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, MIT Press, Cambridge, Massachusetts, 2022.
2. D. S. Hochbaum and J. Naor, "Simple and fast algorithms for linear and integer programs with two variables per inequality", *SIAM Journal of Computing*, 1994, Vol. 23, No. 6, pp. 1179–1192.
3. S. K. Lahiri and M. Musuvathi, "An efficient decision procedure for UTVPI constraints", *Proceedings of the International Workshop on Frontiers of Computing Systems ProCos 2005*, Springer Lecture Notes in Artificial Intelligence, 2005, Vol. 3717, pp. 168–183.
4. K. Subramani1 and P. Wojciechowski, "Integer feasibility and refutations in UTVPI constraints using bit-scaling", *Algorithmica*, 2023, Vol. 85, pp. 610–637.
5. R. J. Vanderbei, *Linear Programming: Foundations and Extensions*, Springer, New York, 2014