$\{0,1\}$ - INTEGER PROGRAMMING WITH COMPLETE CONSTRAINT MATRICES

Jérémi Do Dinh

Abstract

Integer Programming is a well-explored field in Computer Science and Mathematics. Results are often obtained for specific types of integer programs, such as standard forms. Our study imposes additional constraints by considering a complete constraint matrix, which is fixed for a given input size. This specificity provides opportunities for analyzing more tailored solutions and investigating how the problem performs with varied inputs. Our research employs a diverse set of methods to comprehend how to solve the problem and determine feasibility ranges based on its instance.

1 Introduction

General Integer Program feasibility, can be seen as decision problems with inputs corresponding to a matrix $A \in \mathbb{Z}^{m\times n}$ and a right hand side $b \in \mathbb{Z}^m$, where the decision is about whether there exists a binary vector x such that $Ax = b$. Solving such programs requires us to further output a valid solution. The problem explored here, yields many analogies to the general case, however with an additional restriction imposed on the matrix A, by having it to be the complete matrix. The specific nature of our problem is described below.

1.1 The Complete Matrix

In our problem, the notion of the Complete Matrix is essential, and therefore we give its definition.

Definition 1. *Given an integer $m > 0$, we define the complete matrix $\mathbf{A}_m \in \{0,1\}^{m\times 2^m}$ to be the binary matrix, whose columns represent all the distinct binary strings on m bits.*

An example for $m = 5$ is shown below:

$$\begin{bmatrix}
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1\\
0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1\\
0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1\\
0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1\\
0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1
\end{bmatrix}.$$

While it is not necessary to have the ordering of the columns be strictly defined, the matrix above is constructed recursively, for any m, by obtaining two copies of the matrix for $m-1$, appending a row of 0s and 1s at the top of each copy and then putting them side by side, as follows:

$$\mathbf{A}_m = \begin{bmatrix} \mathbf{0}^\top & \mathbf{1}^\top \\ \mathbf{A}_{m-1} & \mathbf{A}_{m-1} \end{bmatrix}.$$

This is represented by the colored squares in the example for $m = 5$ above.

1.2 Problem Statement

Our problem is defined as follows. Given a vector $b \in \mathbb{N}^m$, referred hereinafter as the right-hand side (or RHS), we are interested in obtaining a solution $x \in \{0,1\}^{2^m}$, such that

$$\mathbf{A}_m x = b$$

where $\mathbf{A}_m$ is the complete matrix, following the definition from above. We are also interested in properties and conditions for feasibility of such right hand sides.

2 Preliminaries

2.1 Related Work

The problem at hand is essentially a special form of Integer Program, and we know that general $\{0,1\}$ integer programs[1] are NP-complete [Kar72]. However this is specific to the case where where the input size includes both the size of the matrix $\mathbf{A}$ and the size of the RHS b. Since our problem is well defined based on an input consisting only of the RHS b, we may want to consider other approaches to analyzing the problem.

One idea is to consider how simple modification to the problem at hand can affect where the problem falls in terms of its complexity.

Firstly we notice that restricting ourselves only to a subset of the columns automatically makes the problem NP-hard. The reduction is from the NP-complete ExactSetCover problem [Koz92], where m is defined based on the size of the universe, the columns selected correspond to the family of subsets given as input to the problem, and the right hand side is set to be the $\mathbf{1}$ vector.

On another topic, adding an objective function makes this an Integer Linear Program. Thanks to Eisenbrand and Weismantel [EW20], we know that an Integer Linear Programming problem in standard form:

$$\max\{c^\top x : Ax = b, x \geq 0, x \in \mathbb{Z}^n\}$$

with $A \in \mathbb{Z}^{m \times n}, b \in \mathbb{Z}^m, c \in \mathbb{Z}^n$ can be solved in time $(m \cdot \Delta)^{O(m)} \cdot \|b\|_\infty^2$. Here Δ is the upper bound on each absolute value in A (which in our case is 1). Much of the power of this result comes from the fact that the worst case time doesn't depend on the number of variables (which in our case increases exponentially with the size of the input). In this project, less algorithmic approaches were investigated, with the hope that results can more directly be obtained due to the strict definition of our problem (and in particular the complete matrix).

[1] The problem where given an integer matrix A and an integer vector b, we seek a binary vector x such that $Ax = b$

Our problem can also be made easier, through the consideration of the LP relaxation, where we allow $x \in [0,1]^{2^m}$. Investigating how the fractional solutions relate to integral solutions is interesting both with regards of obtaining the solution and feasibility. We can ask whether a fractional solution necessarily implies an integral one, however this is beyond the scope of this project.

2.2 Intuitions regarding the set of feasible right hand sides

We seek to find a decomposition of each right hand side into a sum of distinct binary vectors of m bits.

Firstly, we might highlight the fact that the ordering of the entries in the right hand side has no implication on the feasibility of the solution. In fact, if we are able to find a solution for a sorted right hand side b_{sort}, we can derive the solution for the original b through a relabeling of variables.

A direct implication of the nature of the complete matrices is that the set $F \in \mathbb{N}^m$ of feasible right hand sides is a subset of $\left[2^{m-1}+1\right]^m$.[2] For $m > 1$, this inclusion is strict. In particular, any vector with all zero entries, except a single entry strictly greater than 1 is infeasible.

This suggests that there are at most $(2^{m-1}+1)^m$ right hand sides. On the other hand we have 2^{2^m-1} linear combinations on the columns of the complete matrix. This puts into perspective the observation that there can be multiple solutions per right hand side.

A somewhat stronger measure is represented by the following lemma.

Lemma 1. *If $b = [b_1, b_2, \dots, b_m]^\top \in \mathbb{N}^m$ is feasible, that is there exists an $x \in \{0,1\}^{2^m}$, such that $\mathbf{A}_m x = b$, then:*

$$\max_i\{b_i\} - \min_i\{b_i\} \leq 2^{m-2}.$$

Proof. Without loss of generality, let's assume that b is sorted in descending order. We claim that b being feasible implies that $b_1 - b_m \leq 2^{m-2}$.

The case where $b_1 \leq 2^{m-2}$ is trivial. Therefore let's suppose that $b_1 > 2^{m-2}$.

Let x^* be a solution for $\mathbf{A}x = b$, and C be the set of columns used to obtain that solution. Let $C^{(1)} \subseteq C$ be the subset of those columns that have a 1 in the top entry. It must hold that $\left|C^{(1)}\right| > 2^{m-2}$. In addition, we can define $\mathbf{A}^{(1)}$ to be the submatrix of $\mathbf{A}$, composed of columns containing 1 in the top entry.

We have that each row of $\mathbf{A}^{(1)}$ contains exactly 2^{m-2} ones, except the first one which contains 2^{m-1} ones. Additionally the elements of $C^{(1)}$ are columns of $\mathbf{A}^{(1)}$. Then since $|C^{(1)}| > 2^{m-2}$, then each row of the sub-matrix of $\mathbf{A}^{(1)}$ formed by the columns in C_1 contains at least $b_1 - 2^{m-2}$ ones. In particular from the definition we have that $b_m \geq b_1 - 2^{m-2}$ and it follows that $b_1 - b_m \leq 2^{m-2}$. □

[2] For $n \in \mathbb{N}^*$, we use the notation $[n]$ for the set $\{0, 1, 2, \dots, n-1\}$

This furthermore restricts the number of possible feasible right hand sides.

3 Main Results

3.1 Maximal Decomposition

One approach to the problem at hand is to look at the maximal decomposition of the right hand sides, which can be the source of patterns for solving the problem. For this, some definitions are useful.

Definition 2. *For a given column c_i of the complete matrix, we define a "pair" to be the tuple (c_i, c_j) such that $c_i + c_j = \mathbf{1}$. We call c_i the complement of c_j and vice-versa.*

Definition 3. *Given a feasible right hand side b we refer to the maximal decomposition of b as the summation of a vector u and the $\mathbf{1}$ vector scaled by a factor $k \in \mathbb{N}$, such that $b = u + k \cdot \mathbf{1}$, where k is maximal such that $\mathbf{A}x = u$ is feasible:*

$$b = u + k \cdot \mathbf{1}.$$

In this case we refer to u as minimal feasible (i.e. $u - \mathbf{1}$ is not feasible).

The set $F_{\min}$ of minimal feasible right hand sides represent yet another subset of all feasible right hand sides, and the ability to check membership in $F_{\min}$ for a right hand side efficiently can be a powerful tool, especially due to the following theorem.

Theorem 2. *Given a maximal decomposition $b = u + k \cdot \mathbf{1}$. If $\mathbf{A}x = b$ is feasible, then each support minimal solution x_u to $\mathbf{A}x = u$ can be extended to a solution $x^* = x_u + x_k$ such that $x_u + x_k \leq \mathbf{1}$.*

Proof. Suppose that this is not true. Therefore there must exist a solution $x' = x_{u'} + x_{k'}$ such that $\mathbf{A}x' = b$, but $x_{u'}$ is not support minimal or k' is not maximal w.r.t. the definition of maximal decomposition and a support minimal solution cannot be extended to a solution of $\mathbf{A}x = b$.

We can define X_1' to be the set of columns used for $x_{u'}$. Based on this set, we define X_2' to be the set of complements of the columns in X_1'. We then let X_R' be the set of remaining columns of $\mathbf{A}$. We note that X_R' can be partitioned into $|X_R'|/2$ pairs.

In the case where $X_1' \cap X_2' \neq \emptyset$, then X_1' must contain a pair (c_i, c_j). In this case, we have that u' can be reduced by $\mathbf{1}$, and thus increase k' by 1, which yields u'' and k''. A new solution to this decomposition can be obtained by setting $x_{u'}^{(i)} = x_{u'}^{(j)} = 0$ to obtain $x_{u''}$ and setting $x_{k'}^{(i)} = x_{k'}^{(j)} = 1$ to obtain $x_{k''}$. This contradicts the maximality of the decomposition.

Let's therefore assume that $X_1' \cap X_2' = \emptyset$. If $k' \leq |X_R'|/2$, then the columns of X_R' are sufficient to define a solution to $\mathbf{A}x = k' \cdot \mathbf{1}$. Then some support minimal solution to $Ax = u'$ would yield $|X_R| > |X_R'|$.

As a result, we can further assume that $k' > |X'_R|/2$. Therefore, there must be a column $c_j \in X'_2$ present in our solution for x'. For this columns there is a complement $c_i \in X'_1$. This can be used to show that u' is not minimal, by defining a solution with a smaller support to solve a smaller u. To achieve this, we set $x_{u'}^{(i)} = 0$, which essentially deletes c_i and c_j from the sets X'_1 and X'_2 respectively and adds them to X'_R. This is followed by setting $x_{k'}^{(i)} = 1$.

This expresses the same solution to $\mathbf{A}x = b$, but using a different decomposition, and using a $u < u'$ that uses less columns. This step can be repeated for every column of X'_2. This way we can obtain a solution that only uses columns of X'_R to obtained the scaled $\mathbf{1}$ vector after we obtain a sufficiently small u''.

It may be that the u'' obtained might not be minimal w.r.t to the maximal decomposition, however if the problem can be solved for some u'' larger than a minimal u, such that the remaining solution uses columns of X'_R, then as already argued this result holds for the maximal decomposition. □

This result is useful especially given the fact that each feasible right hand side can be associated with a unique minimal feasible u. On top of this, we can partition all feasible right hand sides into subsets that are indexed by their minimal feasible u.

Additionally, let's assume that we are given an oracle O_M that on input u can quickly tell us, if u is minimal feasible, and on top of this, another oracle O_S that on input (u, i) can tell us quickly if a support minimal solution to $\mathbf{A}x = u$ uses less than i columns from $\mathbf{A}$. Then, given a right hand side b, we can run the following algorithm to determine feasibility:

```
Input  : $b \in \mathbb{N}^m$, assumed sorted, Oracles $O_M$ and $O_S$
Output: Yes if $b$ is feasible, No otherwise
$m \leftarrow$ len($b$);
if $b_1 - b_m > 2^{m-2}$ then
    return No
end
$b_{\text{cur}} \leftarrow b$;
for $k \leftarrow 0$ to $b_m$ do
    $b_{\text{cur}} \leftarrow b_{\text{cur}} - \mathbf{1}$;
    if $O_M(b_{cur}) =$ Yes and $O_M(b_{cur}, 2^{m-1} - k) =$ Yes then
        return Yes
    end
end
return No
```

In the algorithm above, we simply start from the given RHS b, and iteratively subtract the $\mathbf{1}$ vector. At each step, we make call to the oracle O_M to check if at the given iteration the current vector b_{cur} is minimal feasible. If it is, we make a call to O_M, to check if the support minimal solution to $\mathbf{A}x = b_{\text{cur}}$ uses less than $2^{m-1} - k$. This ensures that there are enough vectors for us to be able to reconstruct the scaled $\mathbf{1}$ vector, to extend the solution of $\mathbf{A}x = b_{\text{cur}}$ to a solution of $\mathbf{A}x = b$.

3.2 Random Informed Guesses

In this section the extent to which we can use randomness to achieve our result was considered. For this we ask some questions about how particular right hand sides, and their nature affect how one can try to solve the problem using a guessing approach.

Namely, if b is feasible, how many times would we have to select a random solution x, until we find a solution to $\mathbf{A}x = b$ if:

1. $\mathbf{1} \cdot 2^{m-3} \leq b \leq \mathbf{1} \cdot 2^{m-2}$
2. $\mathbf{1} \cdot 2^{m/c} \leq b \leq \mathbf{1} \cdot 2^{m-2}$
3. $\mathbf{1} \cdot 2^{m/\log(m)} \leq b \leq \mathbf{1} \cdot 2^{m-2}$

With this in mind, we are interested in finding whether it is possible to define an ordering of feasible right hand sides, such that the number of solutions is non-decreasing.

To tackle these questions, a computer program was created[3] to greedily search for some primary results. To begin with, a search was made through all of the linear combinations for the columns of the matrix, to collect data with regards to how many such combinations map to a particular range. This was done for $m = 4$ and $m = 5$, and the results can be seen in Appendix 1.

To obtain the results for the *Number of linear combinations* column, all of the possible $x \in \{0,1\}^{2^m-1}$ (omitting the $\mathbf{0}$ vector) were enumerated, and multiplied with $\mathbf{A}$. Then it was counted how many such solutions fall within the ranges specified in the leftmost column.

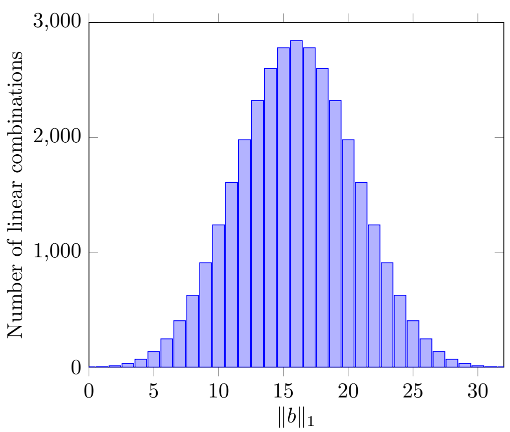
To obtain the results for the *Number of feasible RHS*, the set $\left[2^{m-1}+1\right]^m$ was enumerated, and for each element of this set, the feasibility was checked with the use of the CPLEX module from IBM.

Using these results, it was possible to obtain some statistics such as the average amount of linear combinations per RHS in a particular range, or the proportion of RHS that are in a particular range out of all possible RHS, and similarly for the linear combinations. Those measures are represented in the three entailing columns of the tables.

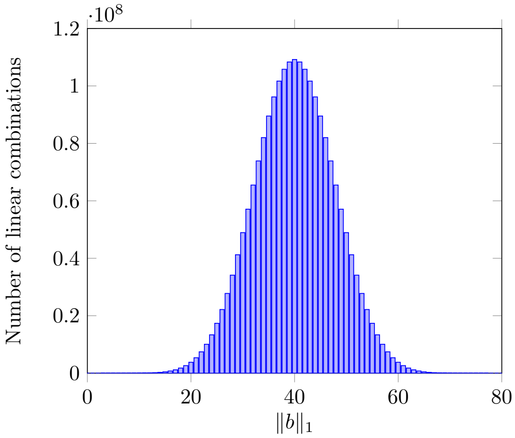
Another measure taken from the enumeration of all possible linear combinations, was a count of the number of solutions that yield a RHS with a particular ℓ_1 norm. Below are the plots for $m = 4$ and $m = 5$:

[3] The code can be found at `https://github.com/jdodinh/CS-498_SemesterProject/tree/main/code`

# of lin. combos. for a given ℓ_1 norm for $m = 4$



# of lin. combos. for a given ℓ_1 norm for $m = 5$



It can be seen from the bar graphs that the highest number of linear combinations yield a right hand side b whose ℓ_1 norm is 16 and 40 for $m = 4$ and $m = 5$ respectively. It may seem that for any m, these values correspond to $m \cdot 2^{m-2}$.

In fact, let $\mathcal{C}(\mathbf{A}_m)$ be the set of columns of $\mathbf{A}_m$. Then if we uniformly and at random pick subsets of columns $C \subseteq \mathcal{C}(\mathbf{A}_m)$, and let our random variable be the ℓ_1 norm of the resulting right hand side, we have that the expected value of this variable would exactly be $m \cdot 2^{m-2}$:

$$\begin{aligned}
\mathbb{E}\left[\|b\|_1\right] &= 2^{-m}\mathbb{E}\left[\sum_{C\subseteq\mathcal{C}(\mathbf{A}_m)} \underbrace{\sum_{c\in C}\|c\|_1}_{\substack{\ell_1 \text{ of RHS} \\ \text{created by } C}}\right] \\
&= 2^{-m}\sum_{C\subseteq\mathcal{C}(\mathbf{A}_m)}\sum_{c\in C}\mathbb{E}\left[\|c\|_1\right] \\
&= 2^{-m}\sum_{C\subseteq\mathcal{C}(\mathbf{A}_m)}\sum_{c\in C}\frac{m}{2} \\
&= 2^{-m}\sum_{c\in\mathcal{C}}\sum_{\substack{C\subseteq\mathcal{C}(\mathbf{A}_m) \\ \text{s.t. } c\in C}}\frac{m}{2} \\
&= 2^{-m}\cdot 2^m\cdot 2^{m-1}\cdot\frac{m}{2} \\
&= m\cdot 2^{m-2}.
\end{aligned}$$

Which is the expected ℓ_1 of the right hand side given the complete matrix, if we were to pick subsets of columns uniformly and at random. This explains why the number of combinations per ℓ_1 norm is symmetrically spread around $m \cdot 2^{m-2}$. The full list of values that yield those charts can be found in Appendix 2.

Unfortunately, coming up with these charts for $m > 5$ is computationally expensive. Since already for $m = 5$, we have 2^{31} linear combination to iterate through, getting a result for any higher value of m is not likely to terminate quickly.

These charts however suggest that based on the ℓ_1 norm of the RHS, we can perhaps direct the search for the columns used, to ideally require less guesses to arrive at our solution.

3.3 Sensitivity

In this section we are interested in knowing whether the ℓ_1 norm of the difference between two distinct right hand sides b and b' can tell us about the difference in the support of their solution. That is whether for all solutions x and x' to $\mathbf{A}x = b$ and $\mathbf{A}x = b'$ respectively, it necessarily holds that $\|x - x'\|_1 \leq p(m)\|b - b'\|_1$ for some function p.

Unfortunately, we have managed to show an example when the sensitivity is large, therefore negating the above argument. Namely that there exists a b and a solution to $\mathbf{A}x = b$, and another rhs b' such that all solutions to $\mathbf{A}x = b'$ are far away from x. We elaborate on the example below.

For any m consider the following right hand side:

$$b = \begin{bmatrix} 2^{m-1} - 1 \\ 2^{m-2} \\ \vdots \\ 2^{m-2} \end{bmatrix}.$$

We can construct a solution $\mathbf{A}x = b$ as follows: choose all the columns from $\mathbf{A}_m$ that have a 1 in the first entry, except the all $\mathbf{1}$ column. This yields a right hand side value of $[2^{m-1} - 1, 2^{m-2} - 1, \cdots, 2^{m-2} - 1]^\top$, and vectors still need to be chosen to satisfy the remaining difference of $\big[0, \underbrace{1, \cdots, 1}_{m-1 \text{ times}}\big]$. To satisfy this we choose all of the $m-1$ remaining unit vectors that aren't yet part of our support.

Now we can provide a new right hand side b' defined as follows:

$$b' = \begin{bmatrix} 2^{m-1} \\ 2^{m-2} \\ \vdots \\ 2^{m-2} \end{bmatrix}.$$

We have that $\|b - b'\|_1 = 1$, and we observe that to satisfy the first entry of this right hand side, we are forced to pick all the columns of $\mathbf{A}_m$ that have a 1 in the first entry. However, picking all of those columns already satisfies b' and furthermore it is the unique solution to b'. Moreover, it means that all of the $m-1$ unit vectors used in our solution to b cannot be in the support. We have that $\|x - x'\| = m$, which provides an example for when the sensitivity is large.

This sensitivity result shows that we cannot necessarily rely on knowledge of results for particular right hand sides to derive a solution to another right hand side that is close with respect to the ℓ_1 norm. In the worst case it is possible to have to change over m columns in the support, which is computationally expensive.

4 Future Work

4.1 Proximity

In this project an interest was given to the proximity of solutions given a right hand side. That is, given b, we would like to find some bound $l_b \leq \|x\|_1 \leq u_b$, for the set of solutions to $\mathbf{A}x = b$ in order to derive how close solutions to the same right hand side may be.

A trivial value for l_b is of course $b_{\max}$, the maximal entry in the right hand side. This follows from the fact, that in order to satisfy the value $b_{\max}$ in the right hand side, we must choose at least $b_{\max}$ columns in our support.

On the other hand, a trivial value for u_b would of course be $\|b\|_1$, because this indicates the worst case scenario where on average for each column c in the support, we would have that $\|c\|_1 = 1$.

There of course exist examples of right hand sides, where these bounds are tight, such as for instance $b = \mathbf{1}$, where we have that $b_{\max} = 1$ and $\|b\|_1 = m$. For those there exist solutions x_l and x_u with $\|x_l\|_1 = 1$ and $\|x_u\| = m$. Namely, for x_l we choose the all 1 column of $\mathbf{A}$, and for x_u we choose the m unit vectors.

The derivation for better proximity bounds is not a subject that was further investigated in this project, and is the reason for which it's mentioned in the scope of future work. We are optimistic that some tighter bound can be derived, that leverage the nature of the RHS in a more thorough manner.

5 Conclusion

In this project, a specific form of Integer Program was investigated. While it is possible to use existing results to solve this problem in an efficient way, the particular nature of the constraints led to an exploration of the topic from different aspects, with the hope that more efficient methods may exist for Integer Programs of this form. This was done through maximal decomposition, a randomized approach as well as analysis of sensitivity and proximity to yield our results. While no concrete solution strategy was derived, those approaches provide valuable insight to the understanding of the problem, which may be useful in the pursuit of deriving an efficient result.

References

[EW20] Friedrich Eisenbrand and Robert Weismantel. Proximity results and faster algorithms for integer programming using the steinitz lemma. *Acm Transactions On Algorithms*, 16(1):5, 2020.

[Kar72] Richard M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations: Proceedings of a symposium on the Complexity of Computer Computations*, pages 85–103. Springer US, Boston, MA, 1972.

[Koz92] Dexter C. Kozen. More np-complete problems. In *The Design and Analysis of Algorithms*, pages 122–127. Springer New York, New York, NY, 1992.

Appendix

1. Results for random informed guesses

$m = 4$

Ranges	# of lin combos	# of feas. RHS	lin. combos/RHS	prop. of feasible RHS	prop. of lin. combos
$2^{m-2} \leq b \leq 2^{m-1}$	11090	457	24.26	0.7312	0.3383
$2^{m-3} \leq b \leq 2^{m-1}$	29230	1099	26.6	0.4577	0.8920
$2^{m-4} \leq b \leq 2^{m-1}$	32300	1410	22.91	0.3442	0.9856
$2^{m-3} \leq b \leq 2^{m-2}$	7887	81	97.37	1	0.2407
$2^{m-4} \leq b \leq 2^{m-2}$	10620	256	41.47	1	0.324
$2^{m-4} \leq b \leq 2^{m-3}$	457	16	28.56	1	0.01395
$0 \leq b \leq 2^{m-1}$	32770	1611	20.34	0.2455	1
$0 \leq b \leq 2^{m-2}$	11090	457	24.26	0.7312	0.3383
$0 \leq b \leq 2^{m-3}$	652	77	8.468	0.9506	0.01990
$0 \leq b \leq 2^{m-4}$	52	16	3.250	1	0.001587

$m = 5$

Ranges	# of lin combos	# of feas. RHS	lin. combos/RHS	prop. of feasible RHS	prop. of lin. combos
$2^{m-2} \leq b \leq 2^{m-1}$	5.464e+08	3.475e+04	1.572e+04	0.5886	0.2545
$2^{m-3} \leq b \leq 2^{m-1}$	2.055e+09	1.094e+05	1.877e+04	0.2948	0.9568
$2^{m-4} \leq b \leq 2^{m-1}$	2.145e+09	1.463e+05	1.466e+04	0.1927	0.9987
$2^{m-5} \leq b \leq 2^{m-1}$	2.147e+09	1.593e+05	1.348e+04	0.1519	0.9999
$2^{m-3} \leq b \leq 2^{m-2}$	4.635e+08	3.125e+03	1.483e+05	1	0.2159
$2^{m-4} \leq b \leq 2^{m-2}$	5.438e+08	1.653e+04	3.289e+04	0.9836	0.2532
$2^{m-5} \leq b \leq 2^{m-2}$	5.463e+08	2.723e+04	2.006e+04	0.8309	0.2544
$2^{m-4} \leq b \leq 2^{m-3}$	1.762e+06	2.430e+02	7.253e+03	1	8.207e-04
$2^{m-5} \leq b \leq 2^{m-3}$	2.179e+06	1.024e+03	2.128e+03	1	1.015e-03
$2^{m-5} \leq b \leq 2^{m-4}$	6995	32	218.6	1	3.257e-06
$0 \leq b \leq 2^{m-1}$	2.147e+09	1.668e+05	1.287e+04	0.1175e-01	1
$0 \leq b \leq 2^{m-2}$	5.464e+08	3.475e+04	1.572e+04	0.5886	0.2545
$0 \leq b \leq 2^{m-3}$	2.233e+06	2780	803.2	0.8896	1.040e-03
$0 \leq b \leq 2^{m-4}$	9736	238	40.91	0.9794	4.534e-06
$0 \leq b \leq 2^{m-5}$	203	32	6.344	1.000	9.453e-08

2. Number of linear combinations with respect to the ℓ_1 norm of the right hand side

$m = 4$

$\|b\|_1$	# of lin. comb.
0	1
1	4
2	12
3	32
4	69
5	136
6	246
7	404
8	626
9	908
10	1238
11	1608
12	1979
13	2320
14	2600
15	2780
16	2842
17	2780
18	2600
19	2320
20	1979
21	1608
22	1238
23	908
24	626
25	404
26	246
27	136
28	69
29	32
30	12
31	4
32	1

$m = 5$

$\|b\|_1$	# of lin. comb.
0	1
1	5
2	20
3	70
4	205
5	552
6	1370
7	3155
8	6875
9	14200
10	27978
11	52890
12	96160
13	168740
14	286530
15	471756
16	754575
17	1174445
18	1781190
19	2635490
20	3808733
21	5381110
22	7438860
23	10069975
24	13357400
25	17372285
26	22165260
27	27757330
28	34132135
29	41228710
30	48936938
31	57096175
32	65498320
33	73893715
34	82002490
35	89529614
36	96180495
37	101680950
38	105794820
39	108339745
40	109201114
41	108339745
42	105794820
43	101680950
44	96180495
45	89529614
46	82002490
47	73893715
48	65498320
49	57096175
50	48936938
51	41228710
52	34132135
53	27757330
54	22165260
55	17372285
56	13357400
57	10069975
58	7438860
59	5381110
60	3808733
61	2635490
62	1781190
63	1174445
64	754575
65	471756
66	286530
67	168740
68	96160
69	52890
70	27978
71	14200
72	6875
73	3155
74	1370
75	552
76	205
77	70
78	20
79	5
80	1