Transformation from ZOE to ILP

Kondaveti Deviram
dk1273
dk1273@rutgers.edu

Chaitanya Chaudhary
cc2264
cc2264@rutgers.edu

Amari Urquhart
awu2
awu2@rutgers.edu

Abstract— Solving ZOEs by transforming it into an ILP can be done by transforming the ZOE problem by adding in certain new constraints. The time complexity for the transformation part is essentially linear in terms of the number of original constraints and the number of variables. The output post processing is minimal as the solution to the transformed ILP problem is in the same format to the ZOEs. This transformation can handle solving 20 variable ZOEs in under a minute.

Index Terms—algorithms, computational complexity, zero-one equations, integer linear programming

I. INPUT FORMAT

Describe what is an expected input in this section.

A Comma Separated Values (CSV) file with rows for constraint's coefficients in the ZOE problem. Following is an example for a 4-variable ZOE

TABLE I
A SAMPLE INPUT CSV FILE

1	0	0	0	=	1
0	1	0	0	=	1
0	0	1	0	=	1
0	0	0	1	=	1

The input CSV file represents the ZOE equations:
x1 = 1, x2 = 1, x3 = 1, x4 = 1

To run the program use the command: python3 Group_8_project.py [Input CSV file]

II. OUTPUT FORMAT

The matrix representation of the transformed ZOE problem (ILP instance), the ZOE instance, and the optimal variable assignments.

This is stored in the output.txt file.

TABLE II
A SAMPLE OUTPUT TXT FILE

Matrix A (coefficients of x_i) and Vector b:

x1	x2	x3	x4	b
1	0	0	0	1
-1	0	0	0	-1
0	1	0	0	1
0	-1	0	0	-1
0	0	1	0	1
0	0	-1	0	-1
0	0	0	1	1
0	0	0	-1	-1
1	0	0	0	1
-1	0	0	0	0
0	1	0	0	1
0	-1	0	0	0
0	0	1	0	1
0	0	-1	0	0
0	0	0	1	1
0	0	0	-1	0

Zero-one equations:
(1 * x1) == 1
Solution for this group: (1, 0, 0, 0)

Zero-one equations:
(1 * x2) == 1
Solution for this group: (0, 1, 0, 0)

Zero-one equations:

(1 * x3) == 1
Solution for this group: (0, 0, 1, 0)

Zero-one equations:
(1 * x4) == 1
Solution for this group: (0, 0, 0, 1)

Overall solution:
x_1 = 1
x_2 = 1
x_3 = 1
x_4 = 1

In the output.txt file, first it stores the values of matrix A and vector b as in the csv file followed by negation of the values as the constraints are always positive. For example, x1 ≥ 1 and x1 ≤ 1 so we need to convert the second equation by multiplying with -1. So, we now have two equations, x1 ≥ 1 and -x1 ≥ 1. Also, added variable constraints -xi ≥ 0 and xi ≥ 1 for each value of i in the txt file.

III. TRANSFORMATION

The transformation algorithm to reduce ZOEs to ILPs involves converting the zero one equations to inequalities and adding in the variable constraints to finally transform it into a standard form of ILP.

A. Optimizing via creating subproblems

To solve the ILP we first assign each constraint to a *group* in the following manner: assign a constraint to a group if that constraint has at least one variable common to that group. This function will iterate through each column (variable) in order to assign constraint functions to groups. For example, a series of constraint functions that use variables x_1 and x_2 will be grouped and a series of constraint functions that use variables x_3 and x_4 will be grouped separately, if they do not share. This linearly increasing function will potentially increase the algorithm's efficiency if it is able to solve via segregating the larger problem into smaller pieces that you can then use to solve the bigger piece. Each group that is created can be reiterated as a subproblem that contains less variables than the original input. When you solve the subproblems, it is locked in and will not change, potentially reducing the amount of iterations that need to occur.

B. Solving ILP

The program utilizes brute force with constraint optimization in order to iterate through potential combinations of the zero-one equation. For each group G_i that is identified through the subproblem optimization, iterate through all possible combinations of the variables in G_i. When the FIRST suitable combination that satisfies all constraint functions within the optimized G_i is found, the group's iteration returns this combination, and the process is continued, iterating through each G_i in the same manner.

This process will in the worst case have 2^n time complexity, indicating a pure brute force of combinations. An average case is hard to determine but optimization can reduce the potential runtime to a more gradual polynomial runtime if groups with less overlap are able to be found. Overall time complexity with C iterations through each column and R iterations through each row to determine constraint groups and then 2^n iterations through 2 options for n variables, leaving $O(2^n + R*C)$.

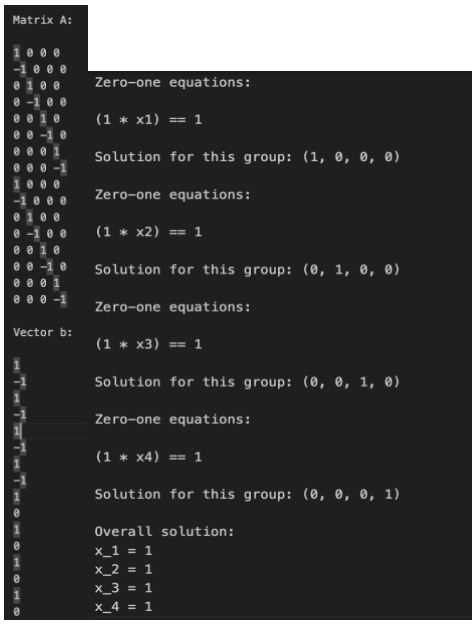
IV. SAMPLE INPUT AND OUTPUT

Example - 1: (Sample input)

Input:

	A	B	C	D	E	F
1	1	0	0	0	=	1
2	0	1	0	0	=	1
3	0	0	1	0	=	1
4	0	0	0	1	=	1

Output:



Example - 2(Solution exists) :

Input:

	A	B	C	D	E	F	G	H	I
1	1	1	0	0	0	0	0	=	1
2	0	0	0	0	0	1	1	=	1
3	0	1	1	1	0	0	0	=	1
4	0	0	1	0	1	0	0	=	1
5	1	0	0	0	1	0	0	=	1
6	0	1	1	0	0	0	0	=	1
7	0	0	0	1	1	0	0	=	1
8	0	0	0	0	0	1	0	=	1

Example - 3(No solution):

Input:

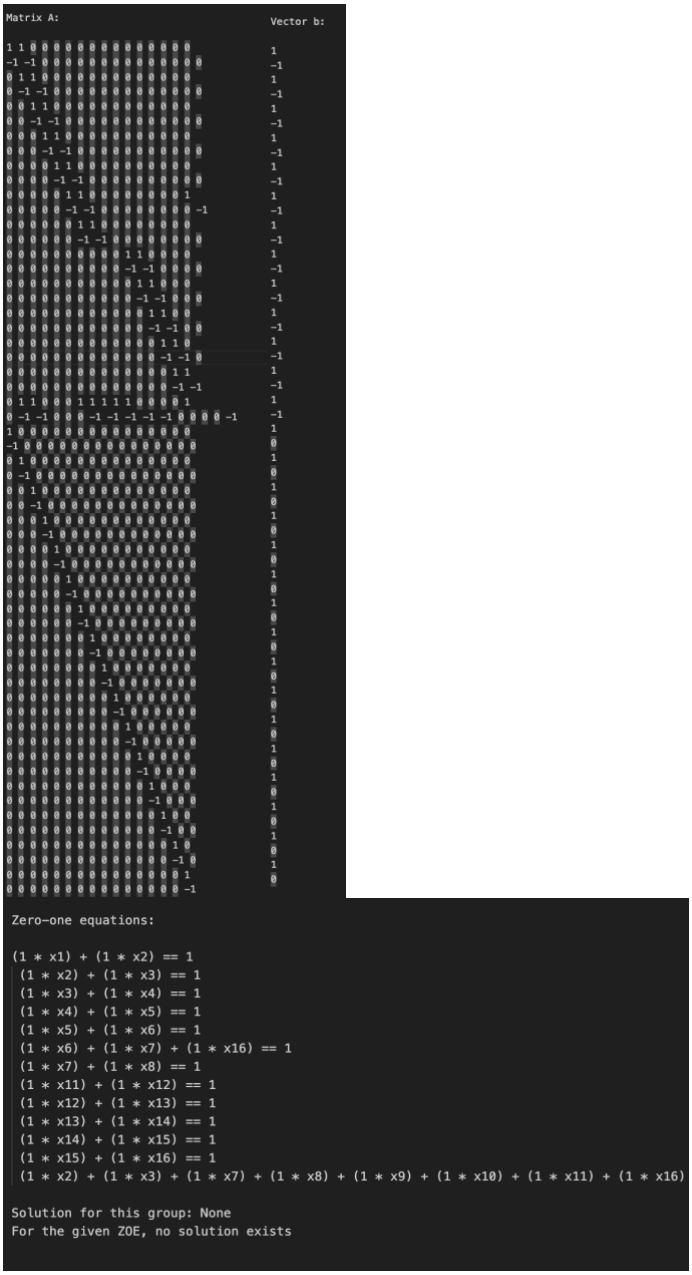
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	=	1
2	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	=	1
3	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	=	1
4	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	=	1
5	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	=	1
6	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	1	=	1
7	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	=	1
8	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	=	1
9	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	=	1
10	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	=	1
11	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	=	1
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	=	1
13	0	1	1	0	0	0	1	1	1	1	1	0	0	0	0	1	=	1

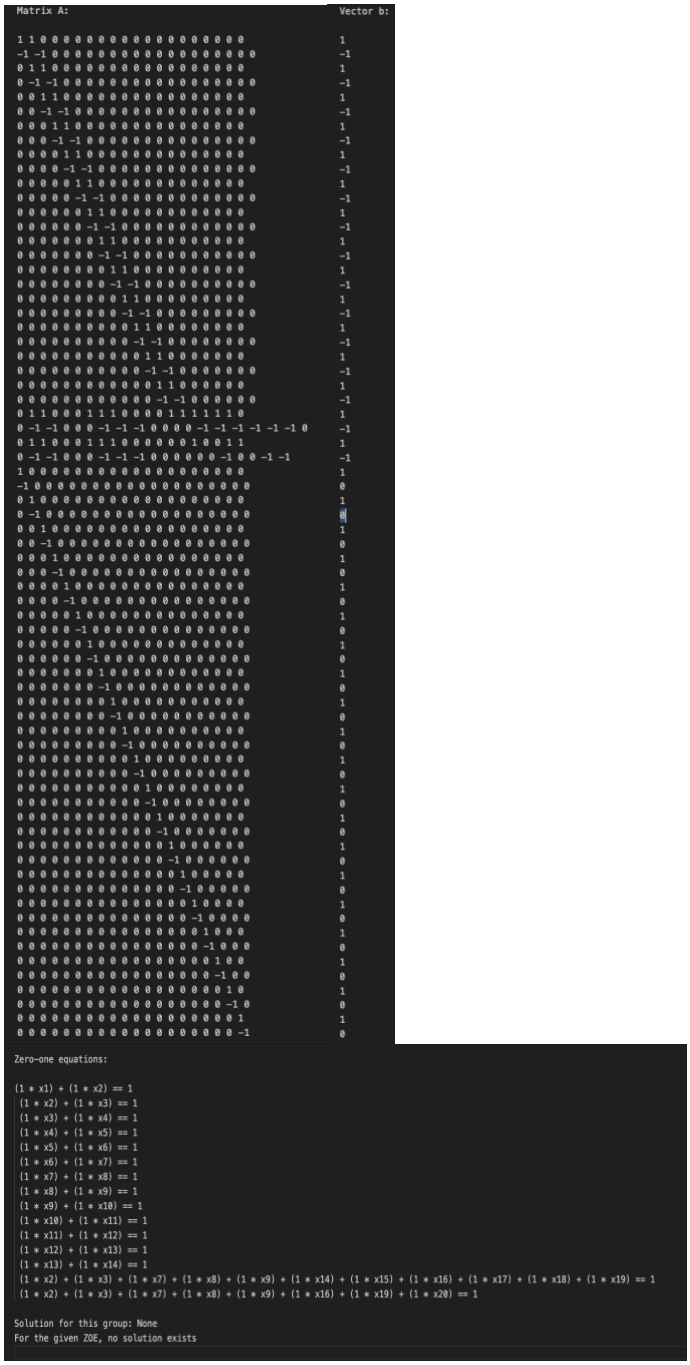
Output:

```
Matrix A:

1 1 0 0 0 0 0
-1 -1 0 0 0 0 0
0 0 0 0 0 1 1
0 0 0 0 0 -1 -1
0 1 1 1 0 0 0
0 -1 -1 -1 0 0 0
0 0 1 0 1 0 0
0 0 -1 0 -1 0 0
1 0 0 0 1 0 0
-1 0 0 0 -1 0 0
0 1 1 0 0 0 0
0 -1 -1 0 0 0 0
0 0 0 1 1 0 0
0 0 0 -1 -1 0 0
0 0 0 0 0 1 0
0 0 0 0 0 -1 0
1 0 0 0 0 0 0
-1 0 0 0 0 0 0
0 1 0 0 0 0 0
0 -1 0 0 0 0 0
0 0 1 0 0 0 0
0 0 -1 0 0 0 0
0 0 0 1 0 0 0
0 0 0 -1 0 0 0
0 0 0 0 1 0 0
0 0 0 0 -1 0 0
0 0 0 0 0 1 0
0 0 0 0 0 -1 0
0 0 0 0 0 0 1
0 0 0 0 0 0 -1

Vector b:

1
-1
1
-1
1
-1
1
-1
1
-1
1
-1
1
-1
1
0
1
0
1
0
1
0
1
0
1
0
```

```
Zero-one equations:

(1 * x1) + (1 * x2) == 1
(1 * x2) + (1 * x3) + (1 * x4) == 1
(1 * x3) + (1 * x5) == 1
(1 * x1) + (1 * x5) == 1
(1 * x2) + (1 * x3) == 1
(1 * x4) + (1 * x5) == 1

Solution for this group: (0, 1, 0, 0, 1, 0, 0)

Zero-one equations:

(1 * x6) + (1 * x7) == 1
(1 * x6) == 1

Solution for this group: (0, 0, 0, 0, 0, 1, 0)

Overall solution:
x_1 = 0
x_2 = 1
x_3 = 0
x_4 = 0
x_5 = 1
x_6 = 1
x_7 = 0
```

Output:





Example - 4(No solution):

Input:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	=	1
2	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	=	1
3	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	=	1
4	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	=	1
5	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	=	1
6	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	=	1
7	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	=	1
8	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	=	1
9	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	=	1
10	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	=	1
11	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	=	1
12	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	=	1
13	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	=	1
14	0	1	1	0	0	0	1	1	1	0	0	0	0	1	1	1	1	1	1	0	=	1
15	0	1	1	0	0	0	1	1	1	0	0	0	0	0	0	1	0	0	1	1	=	1

Output:

V. Programming Language and Libraries Used

List all software tools used.

- Python 3.11
- Numpy 1.26
- Pandas 1.5.3

VI. Conclusions

In conclusion, while the current reduction process from Zero-One Equations (ZOE) to Integer Linear Programming (ILP) is efficient, the solver of ILP does involve iterating through a maximum of 2^16 possible cases for a case with 16 variables and no feasible solution. However, there remains an opportunity for improvement by implementing optimal algorithms to reduce time complexity, ensuring more efficient processing (on average) of larger datasets.

References

[1] Sanjoy Dasgupta, Christos H. Papadimitriou, and Umesh Vazirani. 2006. Algorithms (1st. ed.). McGraw-Hill, Inc., USA.