Linear Inequalities Simplifier
Software Design Document

Name :
Oriel Malihi
Date: (10/01/2021)

TABLE OF CONTENTS

1. INTRODUCTION

1.1 Purpose

This software design document describes the architecture and system design of the linear inequalities simplifier The intended audience for this SDD document are my guider which is PH. D Erel Segal-Halevi, and PH. D Yossi Zaguri which is the Project manager.

1.2 Scope

The main goal is to contribute to the SymPy library, which is an open source python library for computing mathematical calculations, a linear inequalities simplifier. The secondary goal, for me, is to get an experience with connecting and programing with programmers from around the world.

1.3 Overview

To get the a full overview of this document, return one page back to "Table Of Content" section.

1.4 Reference Material

Linear programing:
https://realpython.com/linear-programming-python/

1.5 Definitions and Acronyms

There are few definitions one must know to properly interpret this SDD.

I will list them here:

- **Symbolic computation** - Symbolic computation deals with the computation of mathematical objects symbolically. This means that the mathematical objects are represented exactly, not approximately, and mathematical expressions with unevaluated variables are left in symbolic form.
- **Expression** – mathematical linear expression, such as "3X + 5Y".

- **Inequality** – linear inequality, such as "X + 2Y > 5".
- **'False Set'** - a set of inequalities that at least 2 inequalities contradict each other, for example: "X + Y > 1" and "X + Y < 0.

2. SYSTEM OVERVIEW

There will be 3 functions. The main function will get as input a set of inequalities and one more target inequality and will check if the target inequality is implied from that set. The second function will be an inner function that for a given input of linear inequalities as a set, and an expression, will calculate what is the optional value interval for that expression. If the set of inequalities is a 'false set', the function will return the empty interval no matter what the expression is. The main function using the inner function could determine whether the target inequality is implied from the input set or not. The third function is also an inner function which will get a linear set of inequalities and will simplify it by removing redundant inequalities from this set.

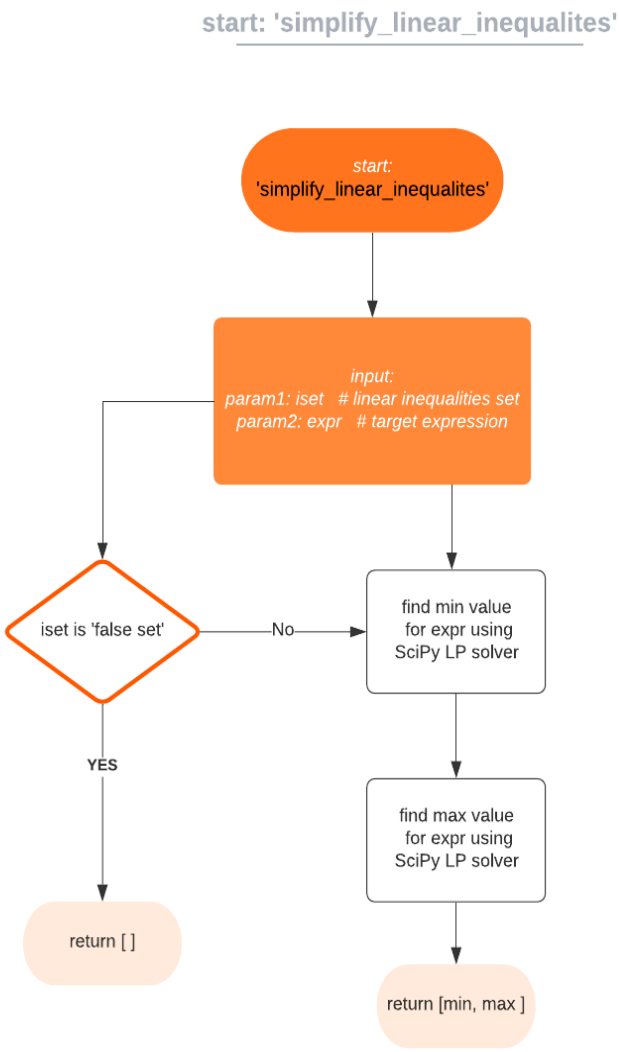
Examples of input/output for these functions can be found here-
https://github.com/orielmalihi/Final-Project/blob/main/iset%20tests.py

3. SYSTEM ARCHITECTURE

3.1 Architectural Design

The main function would be called 'is_implied_by', the second inner function would be called 'find_values_interval'. The third inner function would be called 'simlify_linear_inequalities'. The only subsystem I intend to use is the LP (linear programing) solver from SciPy library. The idea is for a set of inequalities and some target expression to get the min and max values of this expression, using the LP solver, and return the interval [min, max].

3.2 Decomposition



Description

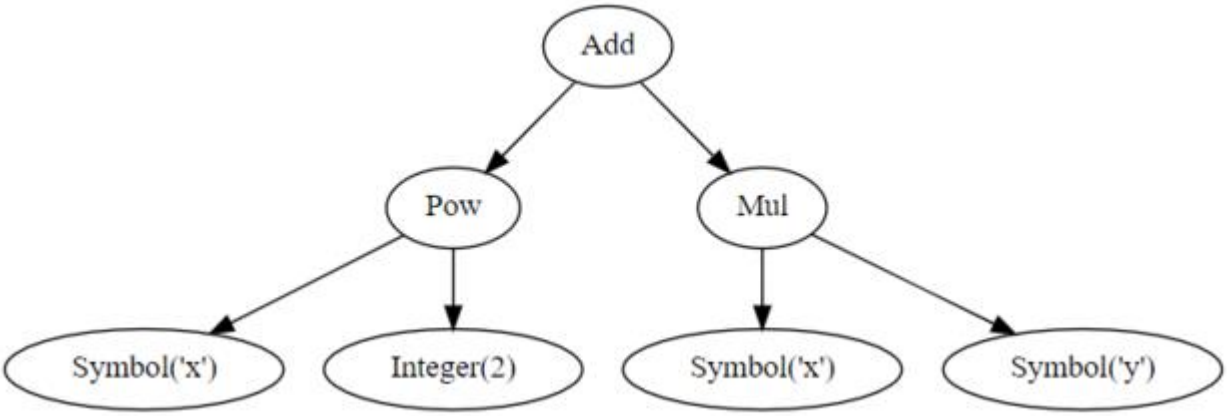
3.3 Design Rationale

I chose to use SciPy and its LP solver since it is probably well implemented and tested, so I can be sure it works well. I also considered to use the LP solver from PuLP library, but I decided not to, because this LP solver uses inequalities as input (rather than matrices of coefficients) and it does not support SymPy's inequalities.

4. DATA DESIGN

4.1 Data Description

In SymPy every mathematical expression is build and stored as a tree. The leefs are symbols and numbers and the other nodes are operation. Note that every sub tree is an expression itself. For example the expression "X^2 + X*Y" is creating this tree-



The inequalities in Sympy are built the same except for two differences. The first one is that every inequalitiy has a unique relation (>, >=, <, <=) and it can not be changed after creation. The second one is that every inequality stores two inner expression, which are called 'lhs' and 'rhs' for the expressions on left and right sides of the inequality. since that in inequality we are often interested in the 2 sides of the relation this is really helpful.

4.2 Data Dictionary

In this project I am going to use SymPy's inequalities, you can see their implementation here- https://github.com/sympy/sympy/blob/master/sympy/core/relational.py

I will list them here with examples:

- GreaterThan – such as "X + Y >= 0".
- LessThan - such as "X + Y <= 0".
- StrictGreaterThan – such as "X + Y > 0".
- StrictLessThan - such as "X + Y < 0".

5. COMPONENT DESIGN

The algorithm is well defined in section 3.2.

As I pointed there I am going to use Scipy's LP solver, which need specific input to solve a linear programing problem. For example, let's say we want to find the optional values of the expression

"X + 2*Y", and we are given by the user the next constraints (which are inequalities), by the algorithm we need to calculate the min and max value of this expression, suppose we want the nax value of the expression, the LP solver would need to solve this problem.

$$\begin{aligned} \text{maximize} \quad & z = x + 2y \\ \text{subject to:} \quad & 2x + y \leq 20 \\ & -4x + 5y \leq 10 \\ & -x + 2y \geq -2 \end{aligned}$$

To solve this linear programing problem the LP solver need to get a matrice of all the variables coefficients (the left sides of the inequalities), a vector numbers which is the right sides of the inequalities, the coefficients of the expression to maximize/minimize (another vector) and bounds for the variables (if there are any bounds).

6. HUMAN INTERFACE DESIGN

6.1 Overview of User Interface

The user will be able to call the 'is_implied_by' function with a set of inequalities as 'constraints' and some target inequality, and the function will return whether this inequality is implied by this set or not.

6.2 Screen Images

Full examples of input/output of those functions can be found here-
https://github.com/orielmalihi/Final-Project/blob/main/iset%20tests.py

7. REQUIREMENTS MATRIX

Since this is an open source contribution project, its requirements are always changing according to the opinion and knowledge of the more senior developers of the library. That is why most of the requirements of the SRS document of this project is irrelevant anymore. And probably many decisions in this document will be changed when the project progresses.