Software Requirements Specification

for

Inequalities Simplifier

Version 1.0 approved

Prepared by Oriel Malihi

Ariel University

10/12/2020

Table of Contents

Table of Contents ..... **ii**
Revision History ..... **ii**
1. Introduction **1**
1.1 Purpose 1
1.2 Intended Audience and Reading Suggestions 1
1.3 Project Scope 1
1.4 References 1
2. Overall Description **2**
2.1 Product Perspective 2
2.2 Product Features 2
2.3 Operating Environment 2
2.4 Design and Implementation Constraints 2
2.5 User Documentation 2
2.6 Assumptions and Dependencies 2
3. System Features **3**
3.1 Simplifiy inequalities ..... **Error! Bookmark not defined.**
3.2 Addition 3
3.3 Subtraction 3
3.4 Check assingment 3
4. External Interface Requirements **4**
4.1 User Interfaces 4
4.2 Software Interfaces 5
5. Other Nonfunctional Requirements **6**
5.1 Software Quality Attributes 6
6. Other Requirements **6**
6.1 **Issues List** **6**

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

Developing an Inequalities Simplifier tool which will allow the user to simplify a set of linear Inequalities to an equivalent set with a smallest number of inequalities if its is possible, or the empty set if it is not.

1.2 Intended Audience and Reading Suggestions

The intended audience for this SRS document are my guider which is PH. D Erel Segal-Halevi, and PH. D Yossi Zaguri which is the Project manager. In both cases I recommend reading the "Purpose" section and than walkthrough the examples specified in the " User Interfaces" and "Hardware Interfaces" sections.

1.3 Project Scope

Developing an Inequalities Simplifier will enable the user to simplify the set of linear Inequalities to a minimized size of equivalent Inequalities which are represented in a symbolic form. Further more the user can add two different sets of Inequalities to a single set or remove from a set a subset of *Inequalities*. he will also be able to check whether a variable assignment to a set of linear Inequalities is valid.

1.4 References

<List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.>

Linear programing:
https://realpython.com/linear-programming-python/
Algorithm for simplifying a set of linear inequalities:
https://or.stackexchange.com/questions/5359/algorithm-for-simplifying-a-set-of-linear-inequalities

2. Overall Description

2.1 Product Perspective

Currently there is no such tool in the SymPy library. Although there is a solve_univariate_inequalities function in SymPy, but it is not good enough since it is limited to the univariate case.

2.2 Product Features

The major features the product contains are simplifying a set of linear Inequalities to a minimized size of equivalent Inequalities which are represented in a symbolic form. the user can add two different sets of linear Inequalities to a single set or remove from a set a subset of Inequalities. The user is also capable of checking whether a variable assignment to a set of linear Inequalities is valid.

2.3 Operating Environment

The environment in which the software will operate is the same environment SymPy currently operates on.

2.4 Design and Implementation Constraints

The most difficulty in this product is to dive into the SymPy library, while learning its code and using it to develop the Inequalities Simplifier. Another problem that might occur is getting the product to fit the SymPy required design patterns.

2.5 User Documentation

The user documentation will be mostly inside the code of the functions in the form of doctest documentation.

2.6 Assumptions and Dependencies

For the developing of this tool we have two assumptions. The first one is that the algorithm for simplifying linear sets of Inequalities (specified in the "References" section) is correct.
The second one is the the linear programing tool in SciPy (specified in the "References" section) is working well without any bugs.

3. System Features

3.1 Simplifying Inequalities

3.1.1 Description and Priority

Simplifying linear sets of Inequalities to an equivalent set with a smallest number of inequalities if its is possible, or the empty set if it is not. High priority feature.

3.1.2 Stimulus/Response Sequences

The user will input the function a set of Inequalities, and the system will output the minimal equivalent set of inequalities.

3.1.3 Functional Requirements

In case of wrong input by the user, such as invalid arguments, the user will get an error so he could try again with a correct form input.

3.2 Addition

3.2.1 Description and Priority

Adding two sets of linear inequalities to a single set. Low medium priority.

3.2.2 Stimulus/Response Sequences

Given two sets of inequalities S1,S2 the user can add one to another by the + operator s.t S2 = S2 + S1.

3.2.3 Functional Requirements

In case of wrong input by the user, such as invalid arguments, the user will get an error so he could try again with a correct form input.

3.3 Subtraction

3.3.1 Description and Priority

Subtracting a subset of linear inequalities from a set of linear inequalities.

3.3.2 Stimulus/Response Sequences

Given a Set S1 with a subset of S2, the user could use the – operator and get S1 minus S2 by typing S1 = S1 – S2.

3.3.3 Functional Requirements

If S2 is not a subset of S1 the user would get an error so he could try again the subtraction with correct form input.

3.4 Check Assignment

3.4.1 Description and Priority

The user could check whether an assignment of the variables is valid for a set of linear inequalities.

3.4.2 Stimulus/Response Sequences

The user would input the assignment to the set of linear inequalities (which will be an object) to a 'check assignment' method and get true if the assignment is valid or false otherwise.

3.4.3 Functional Requirements

For invalid input to the 'check assignment' method, like partial assignment, the user would get an error so he could try again with a correct form input.

4. External Interface Requirements

4.1 User Interfaces

Simplifying inequalities examples:

```
>>> inq1 = x + y ≥ 1
>>> inq2 = x + y < 0
>>> set = simplify_ineq(inq1, inq2)
>>> set
empty
>>>inq2 = 2x + 2y > 3
>>> simplified_set = simplify_ineq(inq1, inq2)
>>> simplified_set
[ 2x + 2y > 3 ]
```

Addition examples:

```
>>> set = simplified_ineq( )
>>> set.add(x + y > 0)
>>>set
```

```
[x + y > 0 ]
>>> set.add(x + y < 10)
>>> set
[ x + y > 0 , x + y < 10 ]
>>> set.add( x + y > 3)
>>> set
[x + y > 3 , x + y < 10]
>>> inq = x + y > 5
>>> set = set + inq
>>> set
[x + y > 5, x + y < 10]
```

Subtraction examples:

```
>>> inq1 = x + y ≥ 1
>>> inq2 = x + y < 7
>>> set = simplify_ineq(inq1, inq2)
>>> set
[x+y ≥ 1, x + y < 7]
>>> set = set - inq1
>>> set
[x + y < 7]
>>> set = set – inq2
>>> set
empty
```

Check Assignment examples:

```
>>> inq1 = x + y ≥ 1
>>> inq2 = x + y < 7
>>> set = simplify_ineq(inq1, inq2)
>>> set.check_assignment({x: 1, y: 2})
True
>>>  set.check_assignment({x: 4, y: 4})
False
```

4.2 Software Interfaces

Given a set of inequalities to the function, the function would run a simplifying algorithm so that any unnecessary inequalities will be removed from the original set.
The algorithm is based on linear programing. For further reading about the algorithm and the linear programing technique you can check the "References" section in this document.

5. Other Nonfunctional Requirements

Software Quality Attributes

The inequalities simplifier should be reliable and give a correct answer for any set of linear inequalities with any number of variables.

6. Other Requirements

Issues List

- How to create Inequalities objects in SymPy.
- How to iterate over the variables of an Inequality object and how to rearrange the Inequality to suit the product goal (like moving a variable from one side of the Inequality to the other side.