Companion Cube Calculator

Geneva Smith

December 17, 2017

1 Revision History

Date	Version	Notes
December 17, 2017	2.0.1	Added a reference to the project repository
December 16, 2017	2.0	Revised the "Composing Operators" instance model and the requirements
October 21, 2017	1.1.1	Collapsed theoretical models for addition, subtraction, and multiplication into one model
October 18, 2017	1.1	Revised table of units and documentation notations
October 4, 2017	1.0	Completed the initial SRS documentation

2 Reference Material

This section records information for easy reference.

2.1 Table of Units

The tasks performed by the C^3 tool are unitless.

2.2 Table of Symbols

The table that follows summarizes the symbols used in this document. The choice of symbols was made to be consistent with mathematical notations of functions, sets, and intervals. The symbols are listed in alphabetical order.

Symbol	Type	Description
a	$\mathbb{R}$	The minimum bound in an interval
b	$\mathbb{R}$	The maximum bound in an interval
$[a, b]$	$[\mathbb{R}, \mathbb{R}]$	Closed interval with endpoints a and b
(a, b)	$(\mathbb{R}, \mathbb{R})$	Open interval with endpoints a and b
B	$\mathbb{R}$	Base number in exponentiation (e.g. B^2)
$D(v)$	$[\mathbb{R}, \mathbb{R}]$	The domain of a variable v represented as a closed, real interval v
$f(x)$	$Type \rightarrow Type$	A mathematical function on a variable x; the function type is determined by the type of x
$f(V)$	$\{[\mathbb{R}, \mathbb{R}]_0, ..., [\mathbb{R}, \mathbb{R}]_n\} \rightarrow [\mathbb{R}, \mathbb{R}]$	A function of closed, real interval domains for a set of variables V that produces a closed, real interval
M	-	A temporary variable for aiding in the readability of equations
$\mathbb{N}$	-	The set of natural numbers that includes 0
N	$\mathbb{N}$	Power in exponentiation (e.g. 2^N)
n	$\mathbb{N}$	A variable subscript denoting placement in an ordered set
op	$\{(Type \rightarrow Type \rightarrow Type)_n\}$	A temporary set of n mathematical operators used to define equation templates; the type of the equation is determined by the type of the input values
$\mathbb{R}$	-	The set of real numbers

Symbol	Type	Description
$R(f(V))$	$[\mathbb{R}, \mathbb{R}]$	The range of $f(V)$ represented as a closed, real interval
V	-	A set of variable names
v	-	A variable name
X	$\{\mathbb{R}_0, ..., \mathbb{R}_n\}$	A closed, connected set of real values
x	$\mathbb{R}$	An element of X
Y	$\{\mathbb{R}_0, ..., \mathbb{R}_n\}$	A closed, connected set of real values
y	$\mathbb{R}$	An element of y
z	$\mathbb{R}$	A temporary value used in operations between X and Y

2.3 Abbreviations and Acronyms

Text	Description
A	Assumption
CRPG	Computer Role-Playing Game
DD	Data Definition
GD	General Definition
GS	Goal Statement
GUI	Graphical User Interface
IM	Instance Model
LC	Likely Change
NPC	Non-Player Character (Games)
PS	Physical System Description
R	Requirement
SRS	Software Requirements Specification
C^3	Companion Cube Calculator
T	Theoretical Model

Contents

1 Revision History **i**

2 Reference Material **ii**
2.1 Table of Units . . . ii
2.2 Table of Symbols . . . ii
2.3 Abbreviations and Acronyms . . . iii

3 Introduction **1**
3.1 Purpose of Document . . . 1
3.2 Scope of Requirements . . . 1
3.3 Characteristics of Intended Reader . . . 2
3.4 Organization of Document . . . 2

4 General System Description **3**
4.1 System Context . . . 3
4.2 User Characteristics . . . 3
4.3 System Constraints . . . 4

5 Specific System Description **5**
5.1 Problem Description . . . 5
5.1.1 Terminology and Definitions . . . 5
5.1.2 Physical System Description . . . 6
5.1.3 Goal Statements . . . 6
5.2 Solution Characteristics Specification . . . 6
5.2.1 Assumptions . . . 6
5.2.2 Theoretical Models . . . 7
5.2.3 General Definitions . . . 8
5.2.4 Data Definitions . . . 9
5.2.5 Instance Models . . . 10
5.2.6 Data Constraints . . . 18
5.2.7 Properties of a Correct Solution . . . 18

6 Requirements **19**
6.1 Functional Requirements . . . 19
6.2 Non-Functional Requirements . . . 19

7 Likely Changes **22**

8 Traceability Matrices and Graphs **23**

3 Introduction

This document is an SRS for the Companion Cube Calculator (C^3), a mathematical tool which determines the range, $R(f(V))$, of a user-specified function, $f(V)$, given the domains of the function's variables, $D(v), v \in V$. This tool will aid in the specification and refinement of GLaDOS, an emotion engine for Non-Player Characters (NPCs) in Computer Role-Playing Games (CRPGs) as described by Smith (2017). The template used to present the required information is based on Smith and Lai (2005); Smith et al. (2007).

The the directory for this project can be found at https://github.com/GenevaS/CAS741.

3.1 Purpose of Document

This document outlines the requirements identified for the development of the C^3 tool, including the product goals, product scope, and the concrete mathematical models driving the design. It also describes the assumptions and theoretical models used to influence the concrete models. The purpose of documenting this information is to aid in future use, maintenance, and development of the C^3 tool.

This document is intended for two reader types – those who wish to use the tool and those who wish to refine and expand the tool. Even though the Companion Cube Calculator was created specifically to aid in the development of the GLaDOS architecture, the user specified $f(V)$ is unitless. This means that the tool can be used for any $f(V)$ as long as any applicable units referred to in $D(v), v \in V$ are compatible.

Since the initial development of the C^3 tool will be limited to arithmetic operators, this document includes information that is useful to a developer looking to expand the abilities of the C^3 with additional mathematical models to suit their project, such as those proposed in LC3 and LC4.

3.2 Scope of Requirements

The C^3 calculates $R(f(V))$ of a user-defined $f(V)$ given the variable domains $D(v), v \in V$. Each $D(v)$ must be specified by the user in the initial version of this tool, otherwise the tool will assume that the user input set is incomplete and treat it as an error.

For the initial version of this product, the mathematical operations allowed in an function will be limited to brackets (()) and the set of mathematical operators $\{+, -, \times, \div, B^x, x^N\}$. Future iterations can expand this list to include trigonometric functions ($\sin(x)$, $\cos(x)$, $\tan(x)$, $\arcsin(x)$, $\arccos(x)$, $\arctan(x)$), partial functions, and other useful mathematical constructs.

Division is a special case in the C^3 tool due to its behaviour around zero. Division by zero is an undefined term, but can be represented by the open interval $(-\infty, \infty)$. This version of the C^3 tool can only process closed intervals (A2), meaning that a divisor interval that contains zero cannot be handled correctly.

The initial version of the tool is intended for use during the development of the GLaDOS architecture. The values calculated by the architecture are normalized, so any uncertainties in the bounds of $R(f(V))$ can be ignored in this context because it is assumed that any calculation errors

will not have a noticeable affect on the architecture's processing. Therefore, directed rounding is not required.

The purpose of this product is to aid in the design and tuning of the GLaDOS architecture (Section 5.1), but it can be used in similar projects because it relies on interval arithmetic and does not contain any GLaDOS-specific concepts or models.

3.3 Characteristics of Intended Reader

The intended reader of this document must understand elementary algebra, especially interval arithmetic, in the domain of real numbers ($\mathbb{R}$). An understanding of set notation and grammar representations are also required to understand the models presented in this document. They must also have an understanding of mathematical domain and range with respect to a function in order to understand the outputs of the product and how it relates to its inputs.

3.4 Organization of Document

This document begins by describing the general description of the C^3 tool (Section 4), which includes the system context, constraints, and the intended users' characteristics. This is followed by a description of the problem to be solved, including the models and assumptions that are used to address it (Section 5). This description is followed by the functional and non-functional requirements of the C^3 tool (Section 6) and suggested refinements and expansions for future iterations (Section 7). The last section in this document, traceability matrices and graphs (Section 8), visually describes the dependencies between different document components. This information can be referenced when making changes to the tool's core specifications.

4 General System Description

This section identifies the interfaces between the system and its environment, describes the user characteristics, and lists the system constraints.

4.1 System Context

The C^3 tool is a stand-alone application for calculating $R(f(V))$ for a user-defined $f(V)$ given the $D(v), v \in V$. Therefore it is independent and self-contained with respect to external organizations, products, and technologies.

The user interface of the C^3 tool facilitates communication between the product and the user, and must contain the ability to exchange user inputs and system outputs. In general, the user is responsible for ensuring that they have provided a semantically correct function for their intended application and that their inputs do not contain unsupported mathematical operations or special functions. The C^3 is responsible for providing an information exchange interface between itself and the user, performing mathematical calculations, and detecting syntactic errors in the system inputs.

- User Responsibilities:
  - Provide $f(V)$ that contains numerical values and supported mathematical operations and symbols
  - Provide $D(v), \forall v \in V$
  - Ensure that $f(V)$ does not contain semantic errors with respect to their intended application
  - Determine if the C^3 outputs are appropriate for the intended application and make adjustments to the program inputs as required
- C^3 Responsibilities:
  - Allow the user to enter $f(V)$ and $D(v), \forall v \in V$ as system inputs
  - Detect data type mismatch, such as a string of characters instead of a floating point number, and communicate this to the user
  - Detect unsupported mathematical operators and bracket mismatches in $f(V)$ and communicate this to the user
  - Detect missing or incomplete $D(v), \forall v \in V$ and communicate this to the user
  - Calculate and output $R(f(V))$ using the user-provided $f(V)$ and $D(v), \forall v \in V$; if a result cannot be calculated, communicate this to the user

4.2 User Characteristics

The end user of C^3 is also the intended reader (Section 3.3) of this document.

4.3 System Constraints

The use of a Graphical User Interface (GUI) is useful for the C^3 tool because it can help the user visualize $f(V)$ during design time and debugging. This has implications on the development languages and target platform selections. If a GUI is included in the C^3 design, only Windows platforms will be supported in the initial release.

5 Specific System Description

This section presents the problem description, which gives a high-level view of the motivation behind the C^3 tool. This is followed by the solution characteristics specification, presenting the assumptions, theories, definitions, and instance models. The solution characteristics are based in interval arithmetic and presented using set notation and language grammars.

5.1 Problem Description

The purpose of this product is to aid in the design and tuning of the GLaDOS architecture, a specialized game engine which enables game designers to create NPCs that react to their environment using models of emotion from psychology. One module in the architecture converts information from the environment into an internal representation that directs an NPC's behaviour selection process. This task requires the specification of several $f(V)$, each with a different variable set V and $D(v), v \in V$. Each $f(V)$ must be normalized to a range of $[-1, 1]$, which can only be done if its $R(f(V))$ is known. The currently implemented $f(V)$ are not well-informed by observation or scientific research and must be subjected to an iterative design process to address this shortcoming. An automated method of calculating $R(f(V))$ for a proposed $f(V)$ will make this process faster and less error-prone.

5.1.1 Terminology and Definitions

This section provides a list of terms and their meaning that are used in the subsequent sections. They are included with the purpose of reducing ambiguity and making it easier to correctly understand the requirements.

- Closed interval: A bounded set that includes its end points; this is expressed $[a, b]$
- Connected set: A set of values that does not contain any disjoint members
- Domain: A connected set of values for a variable x that are valid in $f(x)$
- Extended real interval: The set of all $\mathbb{R}$ that also contains $\pm\infty$
- Mixed interval: An interval that satisfies the property $a \leq 0 \leq b$
- Monotonic function: A function whose first derivative does not change its mathematical sign
- Negative interval: An interval that satisfies the property $a < b < 0$
- Open interval: A bounded set of values that does not contain its end points; this is expressed (a, b)
- Order of Operations: Describes the precedence of mathematical operations when solving an equation (Brackets, Exponents, Division/Multiplication, Addition/Subtraction [BEDMAS])

- Parity: The property of an integer value's inclusion in the even or odd set
- Positive interval: An interval that satisfies the property $0 < a < b$
- Range: The set of values that are produced by a function $f(x)$
- Real interval: A closed, connected set of real numbers

5.1.2 Physical System Description

The C^3 tool does not have a physical system component because it exists independently of the context of $f(V)$.

5.1.3 Goal Statements

Given a user-defined function, which can contain any number of variables, and the minimum and maximum values for each variable in the function, the goal of the C^3 tool is to:

GS1: Determine the minimum and maximum values that the user-defined function can achieve.

5.2 Solution Characteristics Specification

The instance models that govern C^3 tool are presented in Subsection 5.2.5. The information required to understand the meaning of the instance models and their derivation is also presented, so that the instance models can be verified.

5.2.1 Assumptions

This section reduces the scope the original problem to help specify the required theoretical models (Section 5.2.2). The numbers given in the square brackets refer to the theoretical model [T], general definition [GD], data definition [DD], instance model [IM], or likely change [LC], in which the respective assumption is used.

A1: $D(v), \forall v \in V$ is a closed, real interval [DD1, LC1, LC2, LC3, LC4].

A2: $R(f(V))$ is a closed, real interval [DD1, LC1, LC2, LC4].

A3: The mathematical operators used in $f(V)$ can be found in the set $\{+, -, \times, \div, B^x, x^N\}$ [LC4].

A4: Special mathematical functions (e.g. $\sin(x)$, $\cos(x)$, $\log(x)$, ...) are not in $f(V)$ [LC2, LC4].

5.2.2 Theoretical Models

This section focuses on the general equations and laws from interval arithmetic that the C^3 is based on. The models presented here are for real intervals, which can be constrained to closed, real intervals using DD1 to satisfy A1 and A2. Additional models exist for interval arithmetic, but only the models named in A3 are mentioned.

Number	T1
Label	**Addition, Subtraction, and Multiplication on Real Intervals**
Equation	$X < op > Y = \{x < op > y \mid x \in X \wedge y \in Y\}$, where $op \in \{+, -, \times\}$
Description	The summation, subtraction, and multiplication intervals of two real intervals X and Y are equal to the set of sums, differences, and products for each pairwise element x and y.
Source	Hickey et al. (2001)
Ref. By	IM1, IM2, IM3

Number	T2
Label	**Division on Real Intervals**
Equation	$X \div Y = \{z \mid \exists x \in X, y \in Y \wedge y \neq 0, z = x \div y\}$
Description	The quotient interval of two real intervals X and Y is equal to the set of quotients for each pairwise element x and y where $y \neq 0$. If $0 \in Y$, the quotient might not be an interval.
Source	Hickey et al. (2001)
Ref. By	IM4, IM5

Number	T3
Label	**Real Interval Exponents on a Constant Base Number**
Equation	$B^X = \{B^x \mid x \in X \wedge B > 1\}$
Description	The product interval of a constant number $B > 1$ raised to the power of a real interval X is equal to the set of products B^x for each interval element x.
Source	https://en.wikipedia.org/wiki/Interval_arithmetic
Ref. By	IM6

Number	T4
Label	**Constant Exponents on a Real Interval Base Number**
Equations	$X^N = \{x^N \mid x \in X \wedge N \in \mathbb{N}\}$
Description	The product interval of an interval X raised to the power of a constant exponent N is equal to the set of products x^N for each interval element x.
Source	https://en.wikipedia.org/wiki/Interval_arithmetic
Ref. By	IM7

5.2.3 General Definitions

No additional information is required to build the data definitions.

5.2.4 Data Definitions

This section collects and defines all the data needed to build the instance models. No dimensions are required because the tasks performed by C^3 are unitless. These definitions are used to constrain the theoretical models so that they can be translated into instance models on closed, real intervals to satisfy A1 and A2.

Number	DD1
Label	**Representation of a Closed, Real Interval**
Symbol	$[a, b]$
SI Units	-
Equation	-
Description	The definition $[a, b]$ represents a closed, real interval (A1, A2) with endpoints a and b.
Sources	-
Ref. By	IM1, IM2, IM3, IM4, IM5, IM6, IM7

5.2.5 Instance Models

This section transforms the problem defined in Section 5.1 into one which is expressed in mathematical terms. It uses concrete symbols defined in Section 5.2.4 to replace the abstract symbols in the models identified in 5.2.2.

The goal GS1 is solved by recursively solving instance models IM1, IM2, IM3, IM4, IM5, IM6, and IM7 and combining the results (IM8).

Number	IM1
Label	**Addition of closed, real intervals**
Input	$[a_1, b_1]$, $[a_2, b_2]$ from DD1
Output	$[a_{sum}, b_{sum}]$ such that $a_{sum} = a_1 + a_2$, $b_{sum} = b_1 + b_2$, $[a_{sum}, b_{sum}]$ is a closed, real interval (A2)
Description	$[a_1, b_1]$ and $[a_2, b_2]$ are the $D(v)$ for two $v \in V$ in the user's input function $f(V)$. The boundary values a_{sum} and b_{sum} are determined using arithmetic addition. The result, $[a_{sum}, b_{sum}]$, is guaranteed to be a closed, real interval (A2) due to the closed, real interval constraint on $[a_1, b_1]$ and $[a_2, b_2]$ (A1).
Sources	Hickey et al. (2001)
Ref. By	-

Number	IM2
Label	**Subtraction of closed, real intervals**
Input	$[a_1, b_1]$, $[a_2, b_2]$ from DD1
Output	$[a_{sub}, b_{sub}]$ such that $a_{sub} = a_1 - a_2$, $b_{sub} = b_1 - b_2$, $[a_{sub}, b_{sub}]$ is a closed, real interval (A2)
Description	$[a_1, b_1]$ and $[a_2, b_2]$ are the the $D(v)$ for two $v \in V$ in the user's input function $f(V)$. The boundary values a_{sub} and b_{sub} are determined using arithmetic subtraction. The result, $[a_{sub}, b_{sub}]$, is guaranteed to be a closed, real interval (A2) due to the closed, real interval constraint on $[a_1, b_1]$ and $[a_2, b_2]$ (A1).
Sources	Hickey et al. (2001)
Ref. By	-

Number	IM3
Label	**Multiplication of closed, real intervals**
Input	$[a_1, b_1]$, $[a_2, b_2]$ from DD1
Output	$[a_{mul}, b_{mul}]$ such that $a_{mul} = \min(M)$, $b_{mul} = \max(M)$, where $M = \{a_1 \times a_2, a_1 \times b_2, b_1 \times a_2, b_1 \times b_2\}$ $[a_{mul}, b_{mul}]$ is a closed, real interval (A2)
Description	$[a_1, b_1]$ and $[a_2, b_2]$ are the $D(v)$ for two $v \in V$ in the user's input function $f(V)$. The boundary value a_{mul} is determined by taking the minimum of all possible products calculated from the sets $[a_1, b_1]$ and $[a_2, b_2]$, where $[a_1, b_1]$ is the multiplier set and $[a_2, b_2]$ is the multiplicand set The boundary value b_{mul} is determined by taking the maximum of all possible products calculated from the sets $[a_1, b_1]$ and $[a_2, b_2]$, where $[a_1, b_1]$ is the multiplier set and $[a_2, b_2]$ is the multiplicand set The result, $[a_{mul}, b_{mul}]$, is guaranteed to be a closed, real interval (A2) due to the closed, real interval constraint on $[a_1, b_1]$ and $[a_2, b_2]$ (A1).
Sources	Hickey et al. (2001)
Ref. By	-

Number	IM4
Label	**Division of closed, real intervals (Divisor is a positive interval)**
Input	$[a_1, b_1]$, $[a_2, b_2]$ from DD1 where $0 < a_2 \leq b_2$
Output	$[a_{div}, b_{div}]$ such that $a_{div} = a_1 \div b_2, \quad b_{div} = b_1 \div a_2 \quad 0 < a_1 \leq b_1$ $a_{div} = 0, \quad b_{div} = b_1 \div a_2 \quad a_1 = 0 \wedge a_1 \leq b_1$ $a_{div} = a_1 \div a_2, \quad b_{div} = b_1 \div a_2 \quad a_1 < 0 < b_1$ $a_{div} = a_1 \div a_2, \quad b_{div} = 0 \quad b_1 = 0 \wedge a_1 \leq b_1$ $a_{div} = a_1 \div a_2, \quad b_{div} = b_1 \div b_2 \quad a_1 \leq b_1 < 0$ $[a_{div}, b_{div}]$ is a closed, real interval (A2)
Description	$[a_1, b_1]$ and $[a_2, b_2]$ are the $D(v)$ for two $v \in V$ in the user's input function $f(V)$. The boundary values a_{div} and b_{div} are determined by the monotonicity of the interval represented by $[a_1, b_1]$ and whether or not that interval contains 0. The result, $[a_{div}, b_{div}]$, is guaranteed to be a closed, real interval (A2) due to the closed, real interval constraint on $[a_1, b_1]$ and $[a_2, b_2]$ (A1).
Sources	Hickey et al. (2001)
Ref. By	-

Number	IM5
Label	**Division of closed, real intervals (Divisor is a negative interval)**
Input	$[a_1, b_1]$, $[a_2, b_2]$ from DD1 where $a_2 \leq b_2 < 0$
Output	$[a_{div}, b_{div}]$ such that $a_{div} = b_1 \div b_2, \quad b_{div} = a_1 \div a_2 \quad 0 < a_1 \leq b_1$ $a_{div} = b_1 \div b_2, \quad b_{div} = 0 \quad a_1 = 0 \wedge a_1 \leq b_1$ $a_{div} = b_1 \div b_2, \quad b_{div} = a_1 \div b_2 \quad a_1 < 0 < b_1$ $a_{div} = 0, \quad b_{div} = a_1 \div b_2 \quad b_1 = 0 \wedge a_1 \leq b_1$ $a_{div} = b_1 \div a_2, \quad b_{div} = a_1 \div b_2 \quad a_1 \leq b_1 < 0$ $[a_{div}, b_{div}]$ is a closed, real interval (A2)
Description	$[a_1, b_1]$ and $[a_2, b_2]$ are the $D(v)$ for two $v \in V$ in the user's input function $f(V)$. The boundary values a_{div} and b_{div} are determined by the monotonicity of the interval represented by $[a_1, b_1]$ and whether or not that interval contains 0. The result, $[a_{div}, b_{div}]$, is guaranteed to be a closed, real interval (A2) due to the closed, real interval constraint on $[a_1, b_1]$ and $[a_2, b_2]$ (A1).
Sources	Hickey et al. (2001)
Ref. By	-

Number	IM6
Label	**Closed, real intervals as Exponents**
Input	$[a, b]$ from DD1, B
Output	$[a_{exp}, b_{exp}]$ such that $a_{exp} = B^a$, $b_{exp} = B^b$ $[a_{exp}, b_{exp}]$ is a closed, real interval (A2)
Description	$[a, b]$ is the $D(v)$ for a $v \in V$ in the user's input function $f(V)$ which is used as an exponent on a constant value base B. The boundary values a_{exp} and b_{exp} are determined using arithmetic exponentiation. The result, $[a_{exp}, b_{exp}]$, is guaranteed to be a closed, real interval (A2) due to the closed, real interval constraint on $[a, b]$ and the data constraint on B (A1).
Sources	https://en.wikipedia.org/wiki/Interval_arithmetic
Ref. By	-

Number	IM7
Label	**Powers of closed, real intervals**
Input	$[a, b]$ from DD1, N
Output	$[a_{pow}, b_{pow}]$ such that $a_{pow} = a^N, \quad b_{pow} = b^N \quad N$ is odd $a_{pow} = a^N, \quad b_{pow} = b^N \quad N$ is even $\wedge (0 \leq a < b)$ $a_{pow} = b^N, \quad b_{pow} = a^N \quad N$ is even $\wedge (a < b < 0)$ $a_{pow} = 0, \quad b_{pow} = \max(a^N, b^N) \quad$ ELSE $[a_{pow}, b_{pow}]$ is a closed, real interval (A2)
Description	$[a, b]$ is the $D(v)$ for a $v \in V$ in the user's input function $f(V)$ raised to a constant value power N. The boundary values a_{pow} and b_{pow} are determined by the parity of N and the monotonicity of $[a, b]$: • For odd values of N and any interval $[a, b]$, and even values of N when $[a, b]$ is monotonically increasing, $[a_{pow}, b_{pow}]$ are determined by raising each of a and b to the power of N • For even values of N and monotonically decreasing interval $[a, b]$, the boundary values of a and b are reversed before being raised to the power of N • For all other cases, the interval is bounded by 0 and the maximum of the input bounds raised to the power of N The result, $[a_{exp}, b_{exp}]$, is guaranteed to be a closed, real interval (A2) due to the closed, real interval constraint on $[a, b]$ (A1) and the data constraint on N (5.2.6).
Sources	https://en.wikipedia.org/wiki/Interval_arithmetic
Ref. By	-

Number	IM8
Label	**Composing Operations**
Input	$[a_1, b_1], ..., [a_n, b_n]$ from DD1, $f(V)$
Output	$R(f(V)) = [a_R, b_R]$ such that $\langle R(f(V)) \rangle$ ::= $\langle E \rangle$ $\langle E \rangle$ ::= Exp(0) $\langle Exp(p) \rangle$ ::= $\langle P \rangle$ ($\langle B \rangle$ $\langle Exp(q) \rangle$) $\langle P \rangle$ ::= ($\langle E \rangle$) | v $\in [a_1, b_1], ..., [a_n, b_n]$ $\langle B \rangle$ ::= + | - | * | / | ^ $[a_R, b_R]$ is a closed, real interval (A2)
Description	$[a_1, b_1], ..., [a_n, b_n]$ are the $D(v)$ for each $v \in V$ in the user's input function $f(V)$. An $f(V)$ that contains multiple operators can solved recursively using the precedence climbing algorithm. In the production rule for $< Exp(p) >$, the () implies a loop. This loop is will exit when either the next token is a binary operator with an operator precedence that is less than p. If the operator precedence is greater than or equal to p, the precedence value q is calculated as: • $OperatorPrecedence + 1$, if the operator is left associative • $OperatorPrecedence$, otherwise The result, $[a_R, b_R]$, might not be a closed, real interval (A2) due to the mixed interval restriction on divisors (3.2). In this case, the C^3 tool returns an error.
Sources	https://www.engr.mun.ca/~theo/Misc/exp_parsing.htm#climbing
Ref. By	-

5.2.6 Data Constraints

The inputs to the C^3 tool are subject to the following data constraints:

- The function $f(V)$ contains only the mathematical symbols in the set $\{()\} \cup \{+, -, \times, \div, B^x, x^N\}$ (A3), variable names, and constant real values
- If $f(V)$ contains B^x, then $B > 1 \in \mathbb{R}$
- If $f(V)$ contains x^N, then $N \in \mathbb{N}$
- For every variable $v \in V$, $D(v)$ is defined and is a closed, real interval (A1)

The output of the C^3 tool is subject to the following data constraint:

- The interval $R(f(V))$ is a closed, real interval (A2)

5.2.7 Properties of a Correct Solution

The interval of $R(f(V))$ produced by the C^3 tool must exhibit the properties of a closed, real interval (A2). This can be verified mathematically by checking if $R(f(V)) = [a, b]$ satisfies $\{\forall v \in V | a \leq D(v) \leq b \wedge D(v) \in \mathbb{R}\}$.

6 Requirements

This section provides the functional requirements, the business tasks that the C^3 tool is expected to complete, and the non-functional requirements, the qualities that the C^3 tool is expected to exhibit.

6.1 Functional Requirements

R1: The C^3 tool must accept a $f(V)$ and a $D(v)$ for each $v \in V$ from the user. These inputs can be entered directly into the tool or read from a file.

R2: The C^3 tool must convert each $D(v)$ into interval form (DD1).

R3: The C^3 tool must verify that the user inputs satisfy the input data constraints from 5.2.6.

R4: The C^3 tool must decompose $f(V)$ into a series of connected two-operand equations following the standard order of operations rules (BEDMAS) using IM8.

R5: The C^3 tool must solve each equation identified in R4 using IM1, IM2, IM3, IM4, IM5, IM6, and IM7.

R6: The C^3 tool must recompose the equation results from R5 using IM8.

R7: The C^3 tool must verify that the program produces a $R(f(V))$ in interval form (DD1).

R8: The C^3 tool must verify that the program output satisfies the output data constraints from 5.2.6.

R9: The C^3 tool must show the $R(f(V))$ to the user for the given $f(V)$ and $D(v), v \in V$. If this is not possible, such as in the case of a violated data constraint (5.2.6), communicate the reason to the user.

6.2 Non-Functional Requirements

Correctness

- The C^3 tool must be correct in its decomposition of $f(V)$ into smaller equations.
  Fit Criterion: The decomposition process must be proven using mathematical induction.

Reliability

The minimization of floating-point calculation errors is not a priority in the C^3 tool specification. It is predicted that the intended users will be using the tool for tasks that are not sensitive to the inaccuracies produced by floating-point rounding errors.

Robustness

- The C^3 tool must be able to recognize violated data constraints (5.2.6) and report them to the user.
- The C^3 tool must inform the user when it encounters any unspecified state.

Performance

Time and space performance is not a priority in the C^3 tool specification. It is predicted that the intended users will be using the tool in an environment with unrestricted time resources and the space required to perform its calculations will be relatively small in reasonable implementations.

Verifiability

- The C^3 tool must be verifiable with respect to the correctness of its calculations.
  Fit Criterion: The calculation procedures used by the C^3 tool must be implemented such that they can be verified using mathematical proofs.

Usability

- The user must be able to enter $f(V)$ using standard mathematical notation.
- The user must be able to enter values for $D(v), v \in V$ using standard interval notation.
- If the user provides a file as input, they must be able to include both $f(V)$ and each $D(v), v \in V$ in the same file.
- The program must output the value $R(f(V))$ using standard interval notation.
- If the design of the tool includes a GUI (4.3), it must be organized such that it aids in the user's understanding of the order of program inputs.
  Fit Criterion: The intended user should know what inputs are required at each processing stage without referring to the product documentation.

Maintainability

- The evolvability of the C^3 tool must allow the addition of open, real intervals (LC1).
- The evolvability of the C^3 tool must allow the addition of other mathematical operators and special functions (LC3, LC4).

Reusability

Reusability is not a priority because there are currently no future products that will rely on C^3 tool components.

Portability

The portability of the C^3 tool is not a priority because it is expected that the intended users will use Windows platforms for their analysis.

7 Likely Changes

LC1: Support for open, real intervals (A1, A2)

LC2: Determine $D(v)$ for any $v \in V$ if it is not part of the input set (A1, A2)

LC3: Addition of the trigonometric functions $\sin(x)$ and $\cos(x)$ (A1, A4)

LC4: Addition of operators and trigonometric functions whose range includes $\pm\infty$ (A1, A2, A3, A4)

8 Traceability Matrices and Graphs

The purpose of the traceability matrices is to provide easy references on what has to be additionally modified if a certain component is changed. Every time a component is changed, the items in the column of that component that are marked with an "X" may have to be modified as well. Table 2 shows the dependencies of theoretical models, data definitions, and instance models with each other. Items from the likely changes section are not included in this table because implementing the likely changes will require the creation of new models and cannot be adequately addressed by changing the existing models. Table 3 shows the dependencies of instance models, requirements, and data constraints on each other. Table 1 shows the dependencies of theoretical models, data definitions, instance models, and likely changes on the assumptions.

Since there is only one goal statement (GS1) in this specification, it can be assumed that changing the goal will impact every item contained within this document.

	A1	A2	A3	A4
T1				
T2				
T3				
T4				
DD1	X	X		
IM1				
IM2				
IM3				
IM4				
IM5				
IM6				
IM7				
IM8				
LC1	X	X		
LC2	X	X		
LC3	X			X
LC4	X	X	X	X

Table 1: Traceability Matrix Showing the Connections Between Assumptions and Other Items

	T1	T2	T3	T4	DD1	IM1	IM2	IM3	IM4	IM5	IM6	IM7	IM8
T1													
T2													
T3													
T4													
DD1													
IM1	X				X								
IM2	X				X								
IM3	X				X								
IM4		X			X								
IM5		X			X								
IM6			X		X								
IM7				X	X								
IM8					X								

Table 2: Traceability Matrix Showing the Connections Between Items of Different Sections

	IM1	IM2	IM3	IM4	IM5	IM6	IM7	IM8	DD1	5.2.6	R4	R5
IM1												
IM2												
IM3												
IM4												
IM5												
IM6												
IM7												
IM8												
DD1												
R1												
R2									X			
R3										X		
R4								X				
R5	X	X	X	X	X	X	X				X	
R6								X				X
R7									X			
R8										X		
R9										X		

Table 3: Traceability Matrix Showing the Connections Between Requirements and Instance Models

The purpose of the traceability graphs is to provide an alternate visualization of the information presented in the traceability tables. The arrows in the graphs represent dependencies where the component at the tail of an arrow is depended on by the component at the head of that arrow. Therefore, if a component is changed, the components that it points to should also be changed. Figure 1 is a companion to Table 1 and 2, showing the dependencies of theoretical models, data definitions, instance models, likely changes, and assumptions on each other. Diagram elements representing assumption nodes and traces are coloured to aid in its readability. Figure 2 is a companion to Table 3, showing the dependencies of instance models, requirements, and data constraints on each other.

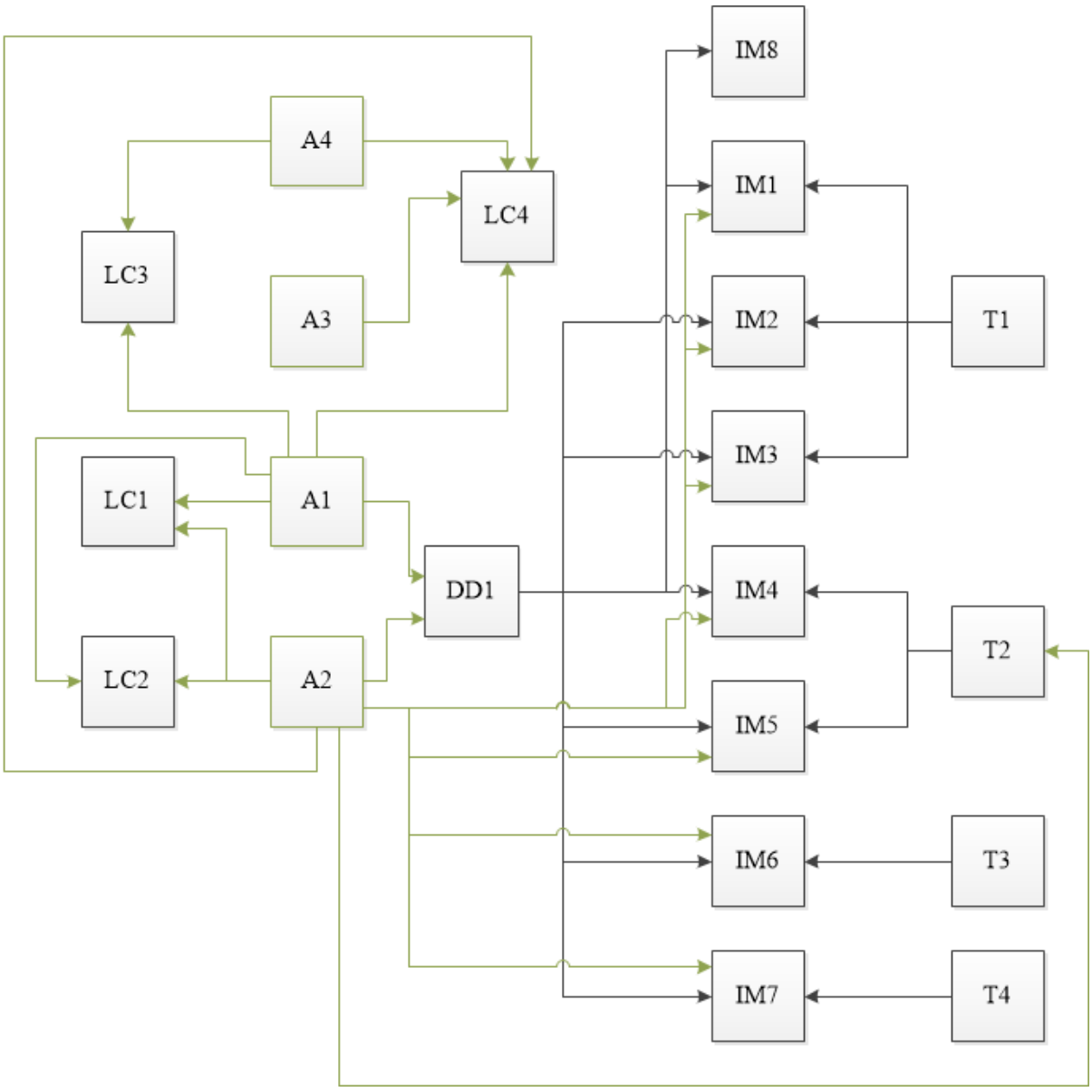

Figure 1: Traceability Matrix Showing the Connections Between Items of Different Sections

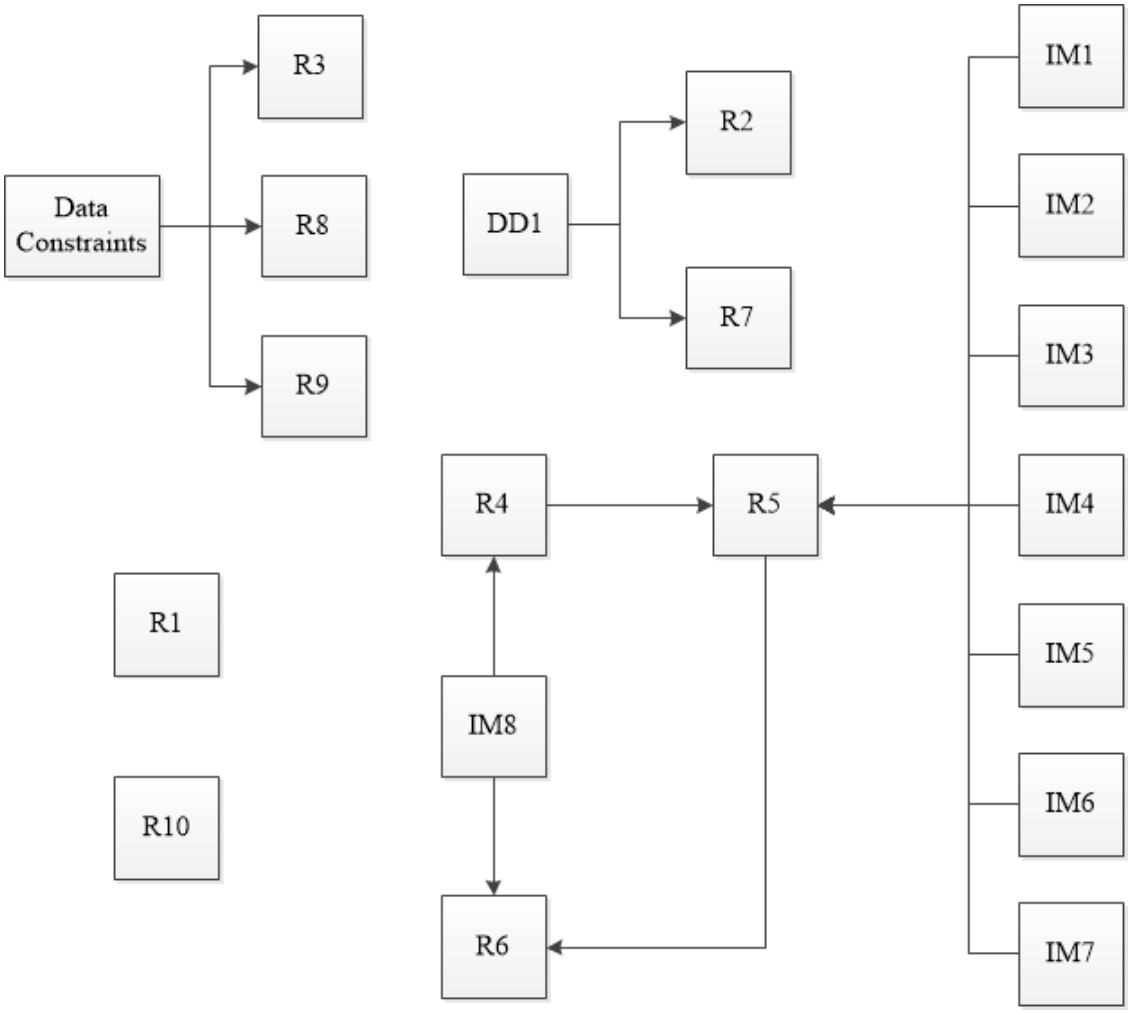

Figure 2: Traceability Matrix Showing the Connections Between Requirements, Instance Models, and Data Constraints

References

T. Hickey, Q. Ju, and M. H. Van Emden. Interval arithmetic: From principles to implementation. *J. ACM*, 48(5):1038–1068, September 2001. URL http://doi.acm.org/10.1145/502102.502106.

Geneva Smith. GLaDOS: Integrating Emotion-Based Behaviours into Non-Player Characters in Computer Role-Playing Games. M.A.Sc. (Software Engineering) Thesis, McMaster University, April 2017. URL http://hdl.handle.net/11375/21369.

W. Spencer Smith and Lei Lai. A new requirements template for scientific computing. In J. Ralyté, P. Àgerfalk, and N. Kraiem, editors, *Proceedings of the First International Workshop on Situational Requirements Engineering Processes – Methods, Techniques and Tools to Support Situation-Specific Requirements Engineering Processes, SREP'05*, pages 107–121, Paris, France, 2005. In conjunction with 13th IEEE International Requirements Engineering Conference.

W. Spencer Smith, Lei Lai, and Ridha Khedri. Requirements analysis for engineering computation: A systematic approach for improving software reliability. *Reliable Computing, Special Issue on Reliable Engineering Computation*, 13(1):83–107, February 2007.