TM Forum Technical Report

Mathematical Functions Definition and Collection

TR292H

Maturity Level: **General availability (GA)**	Team Approved Date: 04-Jul-2024
Release Status: Production	Approval Status: TM Forum Approved
Version 3.6.0	IPR Mode: RAND

Notice

of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.

Direct inquiries to the TM Forum office:

181 New Road, Suite 304
Parsippany, NJ 07054, USA
Tel No. +1 862 227 1648
TM Forum Web Page: www.tmforum.org

Table of Contents

Executive Summary

Mathematical functions are a powerful tool for expression of certain concerns within intent. They allow for example a mapping of values into scores as used when adding utility information to intent. This document defines a set of mathematical functions and vocabulary to manage value ranges and composite functions.

Introduction

Intent requirements are often based on numerical values for properties with continuous value ranges, such as KPI or discrete values such as states. Furthermore, there might be dependencies based on other numerical or discrete concerns. Mathematical functions map input values into output values. They can do this for continuous value ranges or discrete sets of values. This can be very useful in intent to express conditional requirements or supplementary information such as utility that expresses the business value of an observed result. This document introduces mathematical functions as specialization of functions introduced by the function definition ontology TR292C. It provides the vocabulary for specifying custom functions and composite functions. Furthermore, this document pre-defines a collection of useful functions for common use cases in intent expression.

Revision Information

This revision v3.6.0 of the mathematical functions definition and collection model is part of the TM Forum Intent Ontology (TIO) v3.6.0.

The revision v3.6.0 of this document replaces v.3.5.0 with the following changes:

- Minor editorial corrections.

1. Notation and Namespaces

The metrics and observations model is part of the TM Forum Intent Ontology and depends on the following models:

Model	Prefix	Namespace	Published by	Purpose in the model
Mathematical Functions Definition and Collection	mf	http://tio.models.tmforum.org/tio/v3.6.0/MathFunctions/	TM Forum	Identifies the model specified in this document. Definition of mathematical functions
Metrics and observation	met	http://tio.models.tmforum.org/tio/v3.6.0/MetricsAndObservationsSetOperators/	TM Forum	The metrics and observation model introduces metrics as measurable and observable items and defines vocabulary to manage observations about the metric
Set Operators Ontology	set	http://tio.models.tmforum.org/tio/v3.6.0/SetOperators/	TM Forum	Specification of set operators (This document)
Function Definition Ontology	fun	http://tio.models.tmforum.org/tio/v3.6.0/FunctionOntology/	TM Forum	Basic expression of functions
Quantity Ontology	quan	http://tio.labs.tmforum.org/tio/v3.6.0/QuantityOntology	TM Forum	Introduction of quantities and related operators
RDF version 1.1	rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#	W3C	Providing fundamental modeling artifacts
RDF Schema 1.1	rdfs	http://www.w3.org/2000/01/rdf-schema#	W3C	Providing fundamental modeling artifacts
XML Schema	xsd	http://www.w3.org/2001/XMLSchema#	W3C	Providing of data types for literal objects
Time Ontology in OWL	t	http://www.w3.org/2006/time#	W3C	Expression of date and time [owltime]
Examples	ex:	http://www..example.org/	IANA	Reserved domain name for examples

Table 1.1: Model references

The Mathematical Functions Definition and Collection model is based on the Resource Description Framework (RDF) [rdf, rdf_mt, rdf_primer] and the Resource Description Framework Schema (RDFS) [rdfs] published by the World Wide Web Consortium (W3C).

2. Mathematical functions

A mathematical function in the TM Forum intent ontology is expressed by instances of class fun:Function defined in the Function Definition Ontology TR292C. The vocabulary provided by TR292C for defining all aspects of general purpose function apply therefore also to mathematical functions. For example, input parameters and their data types and arity as well as the type of function results would be expressed following TR292C.

The mathematical functions specified in this model use quantities for numerical values. Quantities and quantity operators are defined in the Quantity Ontology TR292D. They represent a value in combination with an optional unit expression. In the context of intent, KPIs and other metrics often require a unit to accompany a numerical value. Therefore, the use of quantities is a sensible choice for mathematical functions in the TM Forum Intent Ontology.

3. Collection of pre-defined functions

This chapter defines a set of functions to be used in intent expression. The chapter contains functions in a two-dimensional space with (x, y) coordinates. The x dimension represents the function input values and y the result values of the function. This means the function maps an x value into a y value.

The functions in this function collection use quantities to represent numerical values if applicable. Numerical literals of type `xsd:integer`, `xsd:decimal` and `xsd:double` can also be used due to their equivalence with quantities without unit.

Each function has a specific implementation based on its shape. It is characterized by a set of parameters. Commonly used parameters in all functions are:

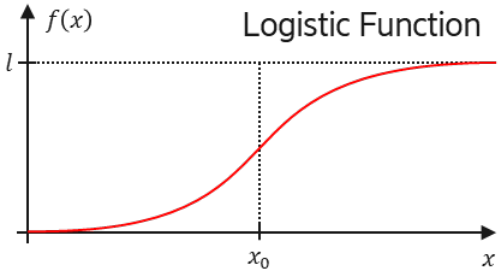
A vertical stretch factor is represented by the property `mf:l`. This is applied by multiplication to the base function and results in stretching the function is y direction.

A vertical offset is represented by the property mf:c. It is applied by addition to the base function and results in a shift in y direction.

A horizontal offset is represented by the property mf:x0. It results in a shift of the function in x direction.

3.1. Logistic function

The function `mf:logistic` represents a logistic function as defined in Figure 1. It is an instance of `fun:function`.



$$f(x) = \frac{l}{1 + e^{-k(x-x_0)}} + c$$

x_0 is the function's midpoint
l is the supremum or upper bound
k is the logistic growth rate or steepness of the curve
c is a vertical offset

Figure 1: Logistic Function

The definition of the logistic function uses the generic properties `mf:l`, `mf:c` and `mf:x0` for general stretching and shifting of the function result. In this respect the property `mf:l` specifies the supremum of the logistics function with a default value of "1". The midpoint is specified by `mf:x0` with the default value of "0". The vertical offset `mf:c` has a default value of "0".

The property mf:k defines the logistic growth rate.

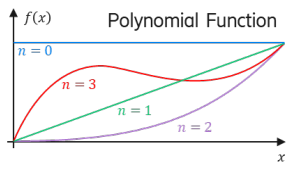
The definition of the logistic function is:

```
mf:logistic
  a fun:function ;
  fun:argumentNames ( mf:input mf:k mf:l mf:x0 mf:c) ;
  fun:argumentTypes ( quan:Quantity ) ;
  fun:resultType quan:Quantity ;
  fun:arityMin "1" ;
.
```

The logistic function defined here allows passing values for the function parameters in its arguments after the input parameter `mf:input`. The result of the logistic function is of type `quan:quantity`.

3.2. Polynomial function

The function `mf:poly` represents a polynomial function as defined in Figure 2. It is an instance of `fun:function`.



$$f(x) = la_n(x - x_0)^n + \cdots + la_2(x - x_0)^2 + la_1(x - x_0) + la_0 + c$$

$$f(x) = l\left(\sum_{k=0}^{n} a_k(x - x_0)^k\right) + c$$

a_n are the function's coefficients
n is the degree of the polynomial

Figure 2: Polynomial Function

The definition of the polynomial function uses the generic properties `mf:l`, `mf:c` and `mf:x0` for general stretching and shifting of the function result. In this respect the property `mf:l` has a default value of "1". The horizontal shift has `mf:x0` has the default value of "0". The vertical offset mf:c has a default value of "0". The vertical offset mf:c is redundant in polynomials as the first coefficient as the same effect. It is kept in the definition of this function for consistency with other functions.

A polynomial function is determined by its coefficients. Any number of coefficients are possible. In this function definition the coefficients are provided as a list starting with the constant coefficient a0 as fist list entry, the linear coefficient a1 as second list entry, the quadratic coefficient a2 as third list entry, etc.

The property `mf:coefficients` specifies the coefficients the polynomial function. Its range is an instance of class `rdf:List`. Coefficients not provided are considered to have the value "0" as default.

The definition of the logistic function is:

```
mf:poly
  a fun:function ;
  fun:argumentNames ( mf:input mf:coefficients mf:l mf:x0 mf:c) ;
  fun:argumentTypes ( quan:Quantity rdf:List quan:Quantity ) ;
  fun:resultType quan:Quantity ;
  fun:arityMin "1" ;
.
```

The polynomial function defined here allows passing values for the function parameters in its arguments after the input parameter `mf:x`. This includes the list of coefficients as second argument. The result of the logistic function is of type `quan:quantity`.

3.3. Discrete Mapping

The function `mf:mapping` represents a mapping function as defined in Figure 3. It is an instance of `fun:function`.

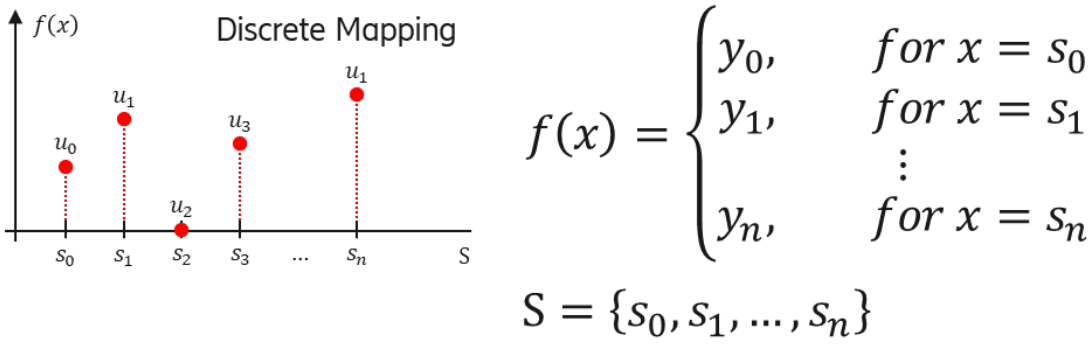


Figure 3: Mapping function

The mapping function is based on explicitly assigning a result to an input value. This is defined with the property `mf:map`.

The range of mf:map is `rdf:List`. This list contains entries that associate a result value with one or several input values.

Each entry of mf:map is a list as well. The first entry of this list is a possible result value. All further entries of this list are input values that shall be mapped into a result represented by the first entry of this list.

Any instance of `rdf:resource` can be specified as input or result. This means literal data types as well as numeric objects such as quantities can be used, but it is not limited to them.

Function provide a unique mapping into results. This means that multiple distinct inputs can lead to the same result, but a single input value cannot have multiple results. The map provided for this function can contain mappings that are not unique. However, the mapping function only maps into the first mapping found that matches the input. It is searching through the map lists in the order of entries.

The first result entry that does not have any specific input associated mapped into it is used as default result. An input that is not mapped explicitly would be mapped into this result.

The definition of the mapping function is:

```
mf:mapping
  a fun:function ;
  fun:argumentNames ( mf:input mf:map )
  fun:argumentTypes ( rdf:Resource rdf:List )
  fun:resultType rdf:Resource ;
  fun:arityMin "1" ;
.
```

For example, the mapping function can be used in the following way:

```
ex:severity [ mf:mapping ( [ ex:rootCause ( ex:incident123 ) ]
                    ( ( ex:Critical ex:Congestion ex:Outage )
                      ( ex:Warning ex:HighLoad )
                      ( ex:Normal )
                    )
                  )
        ]
```

This example uses a mapping function that takes a type of root cause as input and maps it into severity levels. The example function `ex:rootCause` is used here to derive its type from an incident individual. In this example, congestion and outage would be categorized as critical, high load as warning and other possible root cause types as normal.

4. Administrative Appendix

4.1. Document History

4.1.1. Version History

Version Number	Date Modified	Modified by:	Description of changes
3.4.0	29-Feb-2024	Alan Pope	Final edits prior to publication
3.5.0	03-May-2024	Alan Pope	Final edits prior to publication
3.6.0	04-Jul-2024	Alan Pope	Final edits prior to publication

4.1.2. Release History

Release Status	Date Modified	Modified by:	Description of changes
Pre-production	29-Feb-2024	Alan Pope	Initial release
Production	26-Apr-2024	Adrienne Walcott	Updated to reflect TM Forum Approved status
Pre-production	03-May-2024	Alan Pope	Updated to v3.5.0
Production	28-Jun-2024	Adrienne Walcott	Updated to reflect TM Forum Approved status
Pre-production	04-Jul-2024	Alan Pope	Updated to v3.6.0
Production	30-Aug-2024	Adrienne Walcott	Updated to reflect TM Forum Approved status

4.2. Acknowledgments

Team Member (@mention)	Company	Role*
Jörg Niemöller	Ericsson	Author, Project Co-Chair
Kevin McDonnell	Huawei	Project Co-Chair
Yuval Stein	Amdocs	Project Co-Chair
Kamal Maghsoudlou	Ericsson	Key Contributor
Leonid Mokrushin	Ericsson	Key Contributor
Marin Orlić	Ercisson	Key Contributor
Aaron Boasman-Patel	TM Forum	Additional Input
Alan Pope	TM Forum	Additional Input
Dave Milham	TM Forum	Additional Input
Xiao Hongmei	Inspur	Reviewer

Select from: Project Chair, Project Co-Chair, Author, Editor, Key Contributor, Additional Input, Reviewer

5. Appendix A: Vocabulary Reference

This Appendix chapter contains a reference definition of all model vocabulary. It is sorted alphabetically:

5.1. c

The property `mf:c` assigns a vertical offset to a function.

Instance of: `rdf:Property`Domain: `fun:Function`
Range: `quan:Quantity`

5.2. coefficients

The property `mf:coefficients` specifies the list of coefficients that define a polynomial function.

Instance of: `rdf:Property`Domain: `fun:Function`
Range: `quan:Quantity`

5.3. input

The property `mf:input` specifies the input value within a function argument .

Instance of: `rdf:Property`Domain: `fun:Function`

5.4. k

The property `mf:k` specifies the logistic growth of a logistic function.

Instance of: `rdf:Property`Domain: `fun:Function`
Range: `quan:Quantity`

5.5. l

The property `mf:l` assigns a vertical stretch to a function.

Instance of: `rdf:Property`Domain: `fun:Function`
Range: `quan:Quantity`

5.6. logistic

The property `mf:logistic` represents a logistic function.

Instance of: `fun:Function`Argument Names: `_x`, `mf:k`, `mf:l`, `mf:c`
Argument Types: `quan:Quantity`

Result Type: `quan:Quantity`
Arity Min: At least one

5.7. map

The property `mf:map` defines the mapping of input to result values applied in a mapping function. The range is a list with entries that are lists as well which represent individual mapping. The individual mapping lists have the result value as first entry and all other entries are values that are supposed to be mapped into this result.

Instance of: `rdf:Property`Domain: `fun:Function`
Range: rdf:List

5.8. mapping

The property `mf:mapping` represents a mapping function.

Instance of: `fun:Function`Argument Names: `_x`, `mf:map`
Argument Types: `rdf:Resource`, `rdf:List`
Result Type: `rdf:Resource`
Arity Min: At least one

5.9. poly

The property `mf:poly` represents a polynomial function.

Instance of: `fun:Function`Argument Names: `_x`, `mf:coefficients`, `mf:l`, `mf:c`
Argument Types: `quan:Quantity`
Result Type: `quan:Quantity`
Arity Min: At least one

5.10. Vocabulary

The object mf:`Vocabulary` is a container of all model elements.

Instance of: `rdfs:Container`

5.11. x0

The property `mf:x0` assigns a horizontal shift to a function.

Instance of: `rdf:Property`Domain: `fun:Function`
Range: `quan:Quantity`