Minplus-Console V1.5.3 User Manual

Date: 19/11/2021

1 Introduction

The Minplus-Console, also called the Minplus-interpreter, is a tool that allows to perform (Min,+) and (Max,+) Algebra operations for two classes of functions : increasing convex or concave functions and ultimately pseudo-periodic functions, see [1].
The (Min,+)-Algebra is the basis for the Network Calculus formalism (see [2]), which allows to compute upper bounds on end-to-end delays in large scale real-time communication systems such as AFDX and Ethernet TSN networks.

The Minplus-Console is developed by RealTime-at-Work (https://www.realtimeatwork.com). The license of the software is available in Appendix A.

Installer based distributions are available for Windows, Linux and OS X.

2 Changelog

V1.5.3:
- Add maxBacklogPeriod function

V1.5.2:
- Add plot options to remove grid / grey background
- Fix negative coordinates or slopes in upp segments definitions

V1.5.1:
- Extended plot labels features (e.g plot(f, main="Plot with J=" +J))
- Suport of constant as functions in addition and subtraction to function. (e.g f := step(0,5) +1/2)
- Fixed some subtraction and unary minus operations.
- Fixed left-rigth priority for comp operator (f*g comp h = (f*g) comp h)

V1.5.0:
- Support of complex expressions in function definitions (e.g. affine(1+1/3, x*3/4))
- Support of complex expressions using functions values (e.g. affine(f(0), f(1)*2/3))
- Use of '/' for scalar division, 'div' operator is not supported anymore
- Support of function scalar division (e.g. f:=affine(1,1) s:=f/2)
- Support of (max,+) convolution and deconvolution operations, (min, +) is default for operators which are not tagged as (max,+)
- Add option 'out=" to directly save plots to png images
- Do not allow redefinition of a variable changing its type
- Allow line continuation delimiter '\' for readability of scripts
- Introduce ~+ and ~- operators to denote respectively right-hand side and left-hand side limits on a point, e.g. f(0~+)

V1.4.0:
- Post Condition Checker added, see Section 5.2

V1.3.8:
- Correction of several bugs

V1.3.6:
- Compatibility with Java 9 and beyond

3 Usage

3.1 Starting the interpreter

The console may be started in two ways :

1. from the Windows Program Menu (RTaW→Minplus-Console)
2. in a terminal (all OSes), by typing `minplus-interpreter`

The second method has two advantages:

1. the console may be executed from where the script files are located which makes the use of the exec command easier (see Section 6.4)
2. you may specify configuration options, see Section 5.2

3.2 Command-line options

The following table describes the command line options.

Command line argument	Description
`--help`	Print a list of all command line arguments
`--version`	Print the interpreter's version.
`--info`	Print the interpreter's license.
`--func=IMPL`	Set the function class for arrival and service curves: • UPP stands for Ultimately Pseudo Periodic functions (default option). Most precise function class but computations are in general longer. • ICC stands for Increasing and Concave or Convex functions. This class leads to less accurate results but computations are very fast.
`--fraction | -- double`	Exact fraction/double computations. Exact fraction (*i.e.* rational numbers) is default.
`--postchecking`	Activates the post-condition checker. When active, before the console returns a result, some checks are performed regarding expected properties of the result, or relations between inputs and the result.

3.3 Exploring the sample files

On Windows, the sample scripts folder is accessible via the Program Menu :
RTaW→Minplus-Console→Samples
On all platforms, the sample scripts are located under the “Samples” folder in the installation directory.
The samples help to understand the syntax of the available commands (Section 6).

4 Syntax

This section provides an overview of the syntax.

4.1 Functions

Expression	Description
f := FUNCTION_OPERATION	Function definition FUNCTION_OPERATION
f	return the value of f.
f(x)	the value of f at x.
f(x~+)	the right-hand limit of f at x.
f(x~-)	the left-hand limit of f at x.

4.2 Function constructors

Expression	Description
ratency(a,b)	Construct a rate-latency service function with rate a (≥0) and latency b (≥0).
bucket(a,b)	Construct an leaky bucket arrival function with slope a (≥0) and constant b (≥0) and f(0)=0.
affine(a,b)	Construct an affine function with slope a and constant b. The resulting function is right-continuous at x=0, i.e. f(0)=f(0~+).
step(o,h)	Construct a step function with step happening at time o and height h.
stair(o,l,h)	Construct a staircase function with first step at time o, length l and height h.
delay(o)	Construct a burst-delay function that happen at time o.
zero	Construct the function that has zero as value everywhere: f(x)=0 for x≥0.
epsilon	Construct the "epsilon" function: f(x)=+∞ for x≥0.
uaf(SEGMENT+)	Construct an ultimately affine function. The + means that at least one segment is required. The last segment must go until +infinity. See also help(SEGMENT). Example : uaf([(0,-3)1(1,-2)[[(1,-2)2(7,10) [[(7,10)0(+inf,10)[ )
upp([SEGMENT*,] period(SEGMENT*) [, incr [,period]])	Construct an ultimately pseudo-periodic function. The * means optional. "period" is a mandatory field. First segment list is the finite part. The second part is the pseudo-periodic part. The increment is optional. The period is purely informational.

	Example upp(period([(0,0)3/2(2,3)]](2,3)0(5,3)[), 4)
SEGMENT	Examples: • [(x,y)] is the special case for a spot. • [(x1,y1)slope(x2,y2)] • [(x1,y1)slope(x2,y2)[x and y could be any number, or +inf, +infinity, -inf, -infinity.

4.3 Function operations

Expression	Description
f1 /\ f2	Minimum of f1 and f2.
f1 \/ f2	Maximum of f1 and f2.
f1 + f2	Sum of f1 and f2.
f1 - f2	Subtraction of f2 from f1.
f1 * f2	(min,+) convolution of f1 and f2.
f1 / f2	(min,+) deconvolution of f1 and f2.
f1 *_ f2	(min,+) convolution of f1 and f2.
f1 /_ f2	(min,+) deconvolution of f1 and f2.
f1 *^ f2	(max,+) convolution of f1 and f2.
f1 /^ f2	(max,+) deconvolution of f1 and f2.
star(f)	Subadditive closure of f.
hDev(f, g)	Horizontal deviation between f and g
vDev(f, g)	Vertical deviation between f and g
hShift(f, n)	Compute the function identical to f but horizontally shifted by n.
vShift(f, n)	Compute the function which is identical to f but vertically shifted by n.
inv(f)	Compute the pseudo-inverse of f.
upclosure(f)	Compute the "non-decreasing" closure of f.
nnupclosure(f)	Compute the "non-negative decreasing" closure of f.
f comp g	Composition of f and g, *i.e.* for all x, (f comp g)(x) = f(g(x)).
left-ext(f)	Defined as the function g such that for all x, g(x) = f(x~-).

right-ext(f)	Defined as the function g such that for all x, g(x) = f(x~+).
low_inv(f)	Defined as inv(f).
up_inv(f)	Defined as left-ext(inv(f)).
scalar * f	Function multiplication by scalar value.
f * scalar	Function multiplication by scalar value.
f/ scalar	Function division by scalar value.
maxBacklogPeriod(f,g)	Max backlog period length between f and g

4.4 Statements

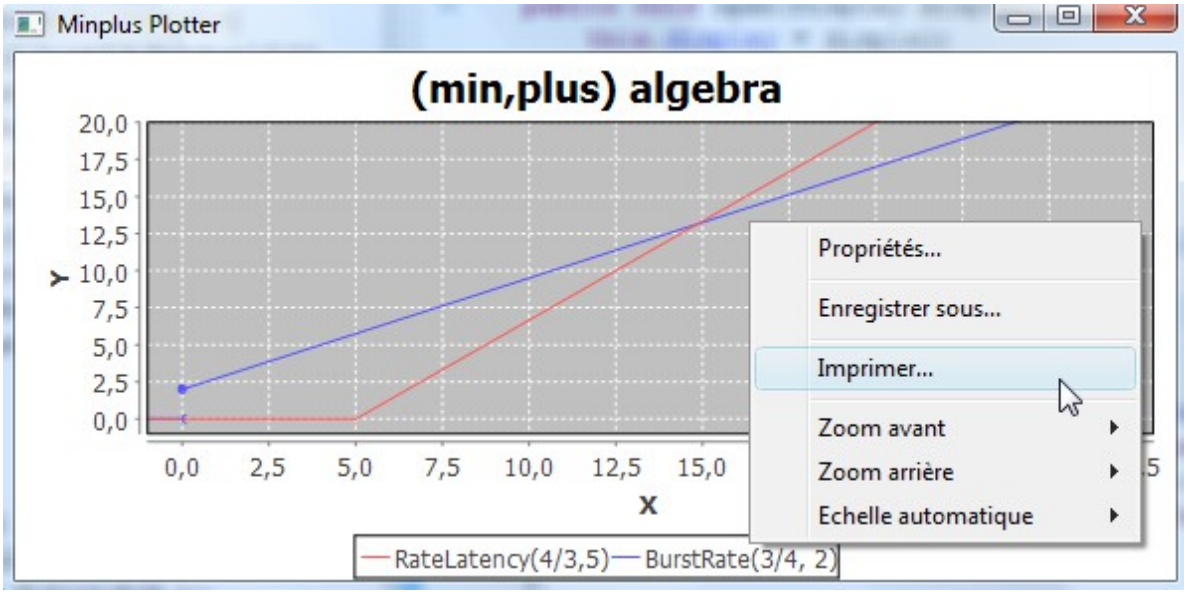
Expression	Description
quit	Exit from the interpreter.
help(fct)	Display help about fct.
exec path	Execute the script located at the given absolute or relative path.
plot(f1,..., args)	Plot a graph displaying the function f1, f2,... args contains parameters for the drawing. Valid args are: • main : the graph title • xlim : range for x-axis • ylim : range for y-axis • xlab : label for x axis • ylab : label for y axis • out : name of png file to save plot to • grid ="no" : remove grid from plot • bg ="no" : use white background instead of default grey Examples: • plot(f1) • plot(f1, f2) • plot(service2,service1,xlim=[-0.3,15],ylim=[-0.3,15]) • plot(f1, main="f1 for J=" +J +"Jitter", xlim=[-0.5, 5], xlab="time", ylab="packets", out = "image.png") • • functions must be variables, they cannot be expressions (e.g., sum of two functions). • Args can be numbers, intervals, string, or string with sum of numbers, variables and strings for labels.
assert(valueExpr1 = valueExpr2)	Assert equality between valueExpr1 and valueExp2
assert(valueExpr1 != valueExpr2)	Assert inequality between valueExpr1 and valueExpr2

assert(valueExpr1 ~= valueExp2)	Assert inequality between valueExpre1 and valueExpr2. Identical to "!=".
assert(valueExpr1 <> valueExpr2)	Assert inequality between valueExpr1 and valueExpr2
assert(functionExpr1 = functionExpr2)	Assert equality between functionExpr1 and functionExpr2
assert(functionExpr1 != functionExpr2)	Assert inequality between functionExpr1 and functionExpr2
assert(functionExpr1 ~= functionExpr2)	Assert inequality between functionExpr1 and functionExpr2
assert(functionExpr1 <> functionExpr2)	Assert inequality between functionExpr1 and functionExpr2
assert(functionExpr1 <= functionExpr2)	Assert functionExpr1 is lower than functionExpr2
assert(functionExpr1 <= functionExpr2)	Assert functionExpr1 is greater than functionExpr2

Note:

Through the context menu (right-click on the plot), plots can be

- exported as PNG images through the "Save As" entry.
- printed through the "Print" entry; if you have a PDF printer installed, you can easily produce PDF images this way.



5 Examples

```
// create three affine functions
f:=affine(7/4,2)
g:=affine(4/5,3)
h:=affine(2/5,5)
// and use them to create an arrival function as their minimum
arrival:=f/\g/\h
// plot the arrival function and the affine functions from which is has been defined
plot(arrival,f,g,h)

// create a "rate latency" service function : we need to take the maximum with
// a horizontal line to obtain the first horizontal segment
service1:=affine(2,-4) \/ affine(0,0)
// plot the service function
plot(service1)

plot(arrival,service1)

// create some other service function
service2:=affine(3.1,-12) \/ affine(0,0)
plot(service2,service1,xlim=[-0.3,15],ylim=[-0.3,15])

// compute the difference between two functions
service3:=service1 - service2
plot(service3,service1,service2)

// compute the convolution of two service functions
convService:=service1*service2
plot(convService,service2,service1)

// compute the convolution of two arrival functions
convArrival:=f*g
plot(convArrival,f,g)

// compute the deconvolution of an arrival function by a service function
deconf:=arrival/service1
plot(deconf,arrival)

// compute the subadditive closure
closure:=star(deconf)
plot(deconf,closure)
```

6 References

[1] *An Algorithmic Toolbox for Network Calculus*, A. Bouillard and E. Thierry. Discrete Event Dynamic Systems, Vol. 18, No. 1, p. 3-49, 2008. Preliminary version available as INRIA Research Report available at url: https://hal.inria.fr/inria-00123643/en/

[2] *Network Calculus*, J.-Y. Le Boudec and P. Thiran, Volume 2050 of LNCS, Springer Verlag, 2001. Available at url: https://ica1www.epfl.ch/PS_files/NetCal.htm

7 Appendix A - End-user license agreement

The software product (hereinafter referred to as the "Product") you are about to install, including its documentation, is protected under the French intellectual property Code. Ownership of this Product is and shall remain vested in REALTIME-AT-WORK (« RTaW »), a French company registered under number 501 464 606, or its licensors.

By installing the Product, you and the entity or company that you represent (hereinafter referred to as the "Customer") acknowledge your understanding and acceptance of the following terms and conditions, and your authority to do so on behalf of your company (if applicable). RTaW is only willing to grant to Customer the right to use the Product under these terms and conditions.
Article 1. License

Subject to the terms and conditions of this Agreement, RTaW hereby grants to Customer, for the number of users, during the term and in the territory, set forth in the applicable order, a non-exclusive and non-transferable license to use the Product and related documentation consisting of RTaW's user manuals and sample files ("Documentation"). Customer's use of the Product (Software and its documentation) shall be limited to internal use.

Customer may not transfer, sell, assign or otherwise convey this Agreement, by operation of law or otherwise, to any other party. Customer may not sell, rent, license, or grant sublicenses, leases, or other rights in the Product to others or otherwise allow the Product to be accessed by another party. Customer may not use the Product to develop, test, support or market competitive products. This Agreement automatically terminates if Customer assigns, sells, rents, licenses, grants, sublicenses, leases or otherwise transfers possession of any copy of the Product to another party or purports to do the same. Except otherwise specified in the order, the maintenance (upgrades and fixes) and support services are not included in the price of the license but can be ordered by Customer separately.

Article 2. Intellectual Property Rights

The Product is a proprietary product of RTaW and is protected by intellectual property law. The Product is deposited at the Swiss "Agence pour la Protection des Programmes" under number IDDN FR 002 020001 00 S P 2015 000 20700. By virtue of this Agreement, Customer acquires only the non-exclusive right to use the Product and does not acquire any rights of ownership or other right in the Product or Documentation. Customer is aware and acknowledges that the Product includes third-party open-source components. RTaW or its licensors shall at all times retain all right, title and interest in the Product and Documentation. Customer acquires no rights of any kind in or to any trade name, logo, or trademark of RTaW.

Article 3. Protection of the Product

Customer shall (i) limit use and disclosure of the Product to its employees and to its consultants who agree to be bound by the terms of this Agreement; (ii) not provide or disclose the Product to any other party without the prior written consent of RTaW; and (iii) take all reasonable precautions to maintain the confidentiality of the Product. Customer agrees, under penalty of license termination but not exclusive of any other remedies, not to cause or permit the reverse engineering, modification, decryption, extraction, disassembly, copying, or decompilation of the Product (except as permitted by applicable law). Customer may copy the Product only for archival purposes. Customer may copy the Documentation (in electronic format) solely for the purpose of facilitating Customer's use of the Product in accordance with, and subject to, the terms and conditions of this Agreement. Customer may not copy nor allow others to copy the Product, or any portion thereof, for any purpose other than expressly set forth herein. Customer agrees not to remove any product identification, copyright notices, or other notices or proprietary restrictions from the Product and may not disclose any information regarding any benchmark or tests of the Product to any third party.

Article 4. Termination

RTaW may terminate this Agreement by written notice if Customer materially defaults in the performance of any provision of this Agreement and fails to cure such default within thirty (30) calendar days after receipt of notice of the default from RTaW. This remedy shall not be exclusive but shall be cumulative upon all other rights and remedies allowed or allowable under this Agreement or by law. On termination of this Agreement for any reason, Customer will destroy all copies of the Product.

Article 5. Warranty - Liability

The Product is provided "as is" without warranty of any kind, either expressed or implied, including, but not limited to, warranty of compatibility with any hardware, software or operating system. RTaW does not warrant the Product will meet Customer's requirements or that its operation will be uninterrupted and error-free.
Customer shall not rely only on the Product to implement a communication architecture which might directly or indirectly jeopardize the safety of people or property.
EXCEPT AS OTHERWISE REQUIRED BY LAW, THE ENTIRE LIABILITY OF RTaW AND CUSTOMER'S EXCLUSIVE REMEDY FOR DAMAGES FROM ANY CAUSE RELATED TO OR

ARISING OUT OF THIS AGREEMENT, WILL NOT EXCEED 50% OF THE SUMS RECEIVED BY RTaW FROM CUSTOMER DURING THE YEAR PRECEDING THE CLAIM.
IN NO EVENT WILL RTaW BE LIABLE FOR ANY LOSS OF REVENUES OR PROFITS, EVEN IF RTaW KNEW OR SHOULD HAVE KNOWN OF THE POSSIBILITY OF SUCH DAMAGES.

Article 6. Miscellaneous

This agreement constitutes the complete agreement between the parties with respect to the Product and supersedes any other agreement, communication or advertising, oral or written, with respect to the Product. THIS AGREEMENT SHALL BE GOVERNED BY AND CONSTRUED UNDER THE LAWS OF FRANCE. IT IS DRAFTED IN ENGLISH AND FRENCH: IN THE EVENT OF DIVERGENCE BETWEEN THE TWO VERSIONS, THE FRENCH DOCUMENT WILL PREVAIL. ANY DISPUTE RELATING TO THE PRODUCT SHALL BE SUBJECT TO THE EXCLUSIVE JURISDICTION OF THE PARIS, FRANCE COURTS. If any provision of this Agreement is held to be unenforceable, such provision shall be limited, modified or severed as necessary to eliminate its unenforceability, and all other provisions shall remain unaffected. RTaW's failure to enforce any provision of this Agreement shall not be deemed a waiver of such provision.
THE COURTS OF PARIS, FRANCE, SHALL HAVE EXCLUSIVE JURISDICTION TO ADJUDICATE ANY DISPUTE ARISING OUT OF OR RELATING TO THIS AGREEMENT. EACH PARTY HEREBY IRREVOCABLY CONSENTS TO THE EXCLUSIVE JURISDICTION OF SUCH COURTS.