FEDERICO SANTA MARÍA TECHNICAL UNIVERSITY
ELECTRONICS DEPARTMENT
VALPARAÍSO - CHILE

Neuronment v1.0

Reference Manual

Version 0.1

Author Pedro F. Toledo
pedrotoledocorrea@gmail.com
Revisers María José Escobar

April 1, 2015

Abstract

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

Glossary

N

Neuronal Network:

TBC
[3]

Neuronment Procedure: NPROC

TBC
[3, 4]

Neuronment Sequencer: NS

TBC
[3, 4]

Neuronment Sequencer Syntax: NSS

TBC
[3, 4]

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Purpose

This document is the Reference Manual for the software Neuronment in his version 1.0 and is intended to provide all the information required for a full usage of its features.

1.2 Style convention

This document uses the following styles:

- **bold**
  Words in bold are used for commands or situations in which they need to be used specifically as indicated including case type.

- *italic*
  Words in italic correspond to words that need to be substituted before using.

- `monospaced`
  Used for examples.

- `<name>`
  In the examples is used for parts that need to be substituted with the real value. Equivalent o italic in normal text.

- `[option]`
  In the examples is used to identify optional arguments.

Bold and italic may also be used to highlight words.

1.3 Problem reporting

If you find a problem, inconsistency or ambiguous explanation please contact the author at pedrotoledocorrea@gmail.com.

1.4 How to read this document

This document is divided in self explanatory chapters divided in 2 groups. Chapters 2, 3 and 4 refer to the common application environment and the following chapters have the instructions and details for the different possible neurological simulations and training procedures.

Chapter 2

User's Guide

2.1 Overview

The Neuronment project is a software intended for discreet simulation and training of complex neural networks for neuroscience studies. Its name comes from the words "Neurological" and "Environment" as the intention is to create a context where different neurological structures can be model, simulated and trained from a simple procedure description file, abstracting all the computational complexities required for the implementation.

Neuronment works by reading a Procedure Description file (standardized extension `*.nproc`) which should describe the list of all the neural network descriptor parameters and the simulations and/or trainings intended to be calculated. This file, also called Neuronment Procedure (NPROC), should comply with the Neuronment Sequencer Syntax (NSS) in order to be correctly interpreted by the Neuronment Sequencer (NS).

The NSS has been developed with the intention to cover all the possible use scenarios for the intended purposes of the Neuronment project; nevertheless, it is susceptible to changes in future versions that may not be backwards compatible.

The Neuronment project has been build mainly on the experience acquire on the development of the thesis work of Pedro F. Toledo[1].

2.2 How to Execute Neuronment

To execute this program it is required the program executable and a "Neuronment Procedure" file. To run it you must execute the following line on the shell:

```
Neuronment -nproc <file.nproc> [-verbose_messages|-no_verbose_messages]
```

- `Neuronment`:
  Name of the Neuronment executable to use.

- `-nproc <file.nproc>`:
  The flag **-nproc** is used to identify the NPROC file that should be read by the neuronment sequencer.

- `verbose_messages no_verbose_messages`:
  This is an optional setting to force the apparition or not apparition of explanatory texts for the coded messages returned by the neuronment sequencer. You can check the default behaviour by checking the private define **DEFAULT_MESSAGES**. CHECK LINKS

2.3 Neuronment sequencer

The neuronment sequencer is the module inside Neuronment on charge of interpreting the NPROC file specified at the program call.

2.3.1 Variable Groups

2.3.2 Message system

2.3.3 Assertions

2.4 Neuronment sequencer syntax

The following section describes the different aspects of the neuronment sequencer syntax required to create a NPROC file.

2.4.1 Basic rules

An NPROC file should be written in ascii and it will be divided in lines using the line breaks as line termination. The resulting lines will be interpreted according their content.

2.4.1.1 Empty line

An empty line prints an empty line to the standard output.

2.4.1.2 Comments

The character `#` divides a line between a command string (everything to the left) and a comment string (everything to the right).

If the command string is only composed of non-script characters, it will be considered empty and the whole line will be interpreted as a comment.

If the whole line is a comment, it will be printed out to the standard output, otherwise, the command will be executed and no comment will be printed to the standard output.

2.4.1.3 Redirections

The character `>` can be used to redirect a command result to a file instead of the standard output.

The text at the right of the `>` character should be only one valid file name without spaces. After the file name it is possible to add a comment as indicated previously.

2.4.1.4 Variables

All interaction related to parameters for the Neuronment operations and the Neuronment environment are managed by variables which value should be set prior the command execution.

Every variable except a “Variable Group” has a parent, by other side every variable could have a child. To specify a variable you should specify the variable group and the variable name separated by a character `:` as the following example:

```
VariableGroup1:Variable1
VariableGroup2:Variable1:ChildVariable1
```

A variable name must be composed exclusively by letters from a to z, A to Z, numbers and the underscore character.

2.4.1.5 Substitutions

If in order to use a command you would like to use a variable value as parameter instead of a hard coded string, you can substitute the name of the variable for its value by using the character `$` just before the variable group as in the following example:

```
$VariableGroup1:Variable1
$VariableGroup2:Variable1:ChildVariable1
```

2.4.1.6 Command results

If a command is intended to return a value after its execution, it will return the values through the Env variable group as it will be indicated in the command specifications.

If you want to store a result you should save it on another variable by using substitution.

Chapter 3

Directives

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

Chapter 4

Messages

The following section includes all the possible interface messages and their description.

4.1 Reading Nproc File Issues

4.1.1 IN-001

- Interface Message:

  NProc directive not recognized

- Development Assertion: YES
- Implementation Assertion: NO
- Runtime Assertion: NO
- Message Description:

  The directive (first word on the nproc command) trying to get interpreted is not on the list of possible directives. Please go to the Reference Manual chapter "Directives" to get a full list of valid directives.

4.1.2 IN-002

- Interface Message:

  Trying to report an undeclared variable

- Development Assertion: YES
- Implementation Assertion: NO
- Runtime Assertion: NO

- Message Description:

  The Simulator as well as the Simulation Environment has a list of predefined variables so store and retrieve information. The variable been addressed is not part of the list. The user is not allowed to create new variables.

4.1.3 IN-003

- Interface Message:

  Redefining a previously stored setting

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: NO
- Message Description:

  A new value has overwritten the value of a previously declared variable.

4.1.4 IN-004

- Interface Message:

  Unidentified sub directive, ignoring line

- Development Assertion: YES
- Implementation Assertion: NO
- Runtime Assertion: NO
- Message Description:

  The sub-directive (second word on the nproc command) trying to get interpreted is not on the list of possible directives. Please go to the Reference Manual chapter ”Directives” to get a full list of valid directives.

4.1.5 IN-005

- Interface Message:

  Unidentified setting, ignoring line

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: NO

- Message Description:

  A value has tried to be written on an un-identified or un-available configuration variable. This nproc line will be ommited

4.1.6 IN-006

- Interface Message:

  Flag not found or without a value

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: NO
- Message Description:

  The command on execution requires the definition of a flag (a word that starts with ”-” in the arguments) that isn’t present or without a value (if a flag requires a value, the next word after it should be the string representing the value for the flag. This string must NOT start with a ”-”. If the value is a negative number put it between quotes).

4.1.7 IN-007

- Interface Message:

  A boolean argument has not been properly written. Will be interpreted as false

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: NO
- Message Description:

  A string been read as boolean doen’t match any of the possible true values (true, True, T, t, 1) neither false (false, False, F, f, 0). It will be interpreted and stored as false.

4.1.8 IN-010

- Interface Message:

  Incorrect amount of arguments for the command, ignoring line

- Development Assertion: NO

- Implementation Assertion: NO
- Runtime Assertion: NO
- Message Description:

  The number of arguments of the command doesn't match the minimum required. This line will be ignored.

4.2 Command Line Issues

4.2.1 UI-001

- Interface Message:

  Duplicated or contradictory flags on command call

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: YES
- Message Description:

  The Neuronment command-line call has a flag declared more than once or two flags that are different are trying to set a contradictory behaivior

4.2.2 UI-002

- Interface Message:

  Flag expected

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: YES
- Message Description:

  The Neuronment command-line call holds a value in a place where should be a flag (a tring starting with ”-”).

4.2.3 UI-003

- Interface Message:

  Flag not recognized, flag omitted

- Development Assertion: NO

- Implementation Assertion: NO
- Runtime Assertion: NO
- Message Description:

  The Neuronment command-line call has detected a flag that isn't on the list of possible flags. This flag will be ignored.

4.2.4 UI-004

- Interface Message:

  Label without content

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: YES
- Message Description:

  The Neuronment command-line call has a flag that requires a value, but the value hasn't been found. If you are trying to use a negative number put it between quotes.

4.2.5 UI-006

- Interface Message:

  Empty NProc name

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: YES
- Message Description:

  The call to Neuronment requires a mandatory nproc file to be processed.

4.3 File IO Issues

4.3.1 ER-001

- Interface Message:

  Development Assetion

- Development Assertion: NO

- Implementation Assertion: NO
- Runtime Assertion: NO
- Message Description:

  The program arrived to an unexpected set of conditions.

4.3.2 ER-002

- Interface Message:

  Runtime Assertion

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: NO
- Message Description:

  Something went wrong running the program. Terminating

4.3.3 ER-003

- Interface Message:

  File couldn't be opened

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: YES
- Message Description:

  The nproc required for execution file couldn't be oppened

4.3.4 ER-004

- Interface Message:

  File couldn't be properly closed

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: NO

- Message Description:

  The nproc file previously executed didn't returned a propper closed status from the OS.

4.3.5 ER-005

- Interface Message:

  Trying to get a new line from a non ready file

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: YES
- Message Description:

  This happens when for some reason the interpreter is trying to get a new line from a file that hasn't been properly oppened

4.3.6 ER-006

- Interface Message:

  Trying to get a new line from a file already at the end

- Development Assertion: YES
- Implementation Assertion: NO
- Runtime Assertion: NO
- Message Description:

  This happens when for some reason the interpreter is trying to get a new line from a file already at the EOF

4.3.7 ER-007

- Interface Message:

  Fail on getting a new line from nproc file

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: YES

- Message Description:

  This happens when for some reason the file stream under interpretation isn't able to retreive a new line from the nproc file, even if it is not at the EOF.

4.3.8 ER-008

- Interface Message:

  Implementation Assertion

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: YES
- Message Description:

  There is a problem or an incompete implementation of a required feature

4.3.9 ER-009

- Interface Message:

  Trying to close an unopened file

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: NO
- Message Description:

  The interpreter is trying to close a file that hasn't been oppened

4.3.10 ER-010

- Interface Message:

  Required file is empty

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: YES
- Message Description:

  The interpreter requires a file for execution. In this case the file name is empty.

4.4 Implementation Issues

4.4.1 DV-001

- Interface Message:

  Missing implementation

- Development Assertion: NO
- Implementation Assertion: YES
- Runtime Assertion: NO

4.4.2 DV-002

- Interface Message:

  Trying to declare a previously declared setting

- Development Assertion: NO
- Implementation Assertion: YES
- Runtime Assertion: NO

4.4.3 DV-003

- Interface Message:

  Trying to load a setting of an unsupported type

- Development Assertion: NO
- Implementation Assertion: YES
- Runtime Assertion: NO

4.4.4 DV-005

- Interface Message:

  Trying to use an unsupported setting

- Development Assertion: NO
- Implementation Assertion: YES
- Runtime Assertion: NO

4.4.5 DV-006

- Interface Message:

  Trying to read an incorrect data type for the setting

- Development Assertion: NO
- Implementation Assertion: YES
- Runtime Assertion: NO

4.4.6 DV-007

- Interface Message:

  The setting count is different than the declared setting count on the HashEntry

- Development Assertion: NO
- Implementation Assertion: YES
- Runtime Assertion: NO

4.4.7 DV-012

- Interface Message:

  Trying a quick access of incorrect type

- Development Assertion: NO
- Implementation Assertion: YES
- Runtime Assertion: NO

4.4.8 DV-013

- Interface Message:

  Hash Table Full

- Development Assertion: NO
- Implementation Assertion: YES
- Runtime Assertion: NO

4.4.9 DV-015

- Interface Message:

  Setting declared but hasn't been initialized

- Development Assertion: NO
- Implementation Assertion: YES
- Runtime Assertion: NO

4.4.10 DV-017

- Interface Message:

  Discrepancy on neuron type on assignment

- Development Assertion: NO
- Implementation Assertion: YES
- Runtime Assertion: NO

4.4.11 DV-018

- Interface Message:

  Trying to quick retrieve un-existent value

- Development Assertion: NO
- Implementation Assertion: YES
- Runtime Assertion: NO

4.4.12 SD-030

- Interface Message:

  Internal simulation pointer corruption

- Development Assertion: NO
- Implementation Assertion: YES
- Runtime Assertion: NO

4.4.13 SD-031

- Interface Message:

  Trying to compare 2 identical MT cells

- Development Assertion: NO
- Implementation Assertion: YES
- Runtime Assertion: NO

4.4.14 SD-032

- Interface Message:

  Empty pointer to function entry

- Development Assertion: NO
- Implementation Assertion: YES
- Runtime Assertion: NO

4.5 Reporting Issues

4.5.1 RP-001

- Interface Message:

  Trying to open an already opened stream

- Development Assertion: NO
- Implementation Assertion: YES
- Runtime Assertion: NO

4.5.2 RP-002

- Interface Message:

  Trying to close a non opened stream

- Development Assertion: NO
- Implementation Assertion: YES
- Runtime Assertion: NO

4.5.3 RP-003

- Interface Message:

  Stream didn't oppened correctly

- Development Assertion: NO
- Implementation Assertion: YES
- Runtime Assertion: NO

4.6 Simulator Interface Issues

4.6.1 SD-001

- Interface Message:

  Simple Simulator Not Initialized

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: YES

4.6.2 SD-015

- Interface Message:

  No V1_Neuron created

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: YES

4.6.3 SD-016

- Interface Message:

  No MT_Neuron created

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: YES

4.6.4 SD-021

- Interface Message:

  Invalid timing for eternal excitation

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: YES

4.6.5 SD-025

- Interface Message:

  Unordered external excitation phase insertion attempt

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: YES

4.6.6 SD-027

- Interface Message:

  First diffusion phase should be always zero

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: YES

4.6.7 SD-028

- Interface Message:

  The number of steps for simulate needs to be at least 1

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: YES

4.6.8 SD-033

- Interface Message:

  Trying to access an invalid activation TimeStep

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: YES

4.6.9 SD-036

- Interface Message:

  Setting not recognized

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: YES

4.6.10 SD-037

- Interface Message:

  Setting required

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: YES

4.7 Warnings

4.7.1 WN-006

- Interface Message:

  There are undocumented calculations in use

- Development Assertion: NO
- Implementation Assertion: NO
- Runtime Assertion: NO

Bibliography

[1] Pedro F. Toledo. *Implementation of a multithreaded numeric genetic algorithm with decreasing mutation impact for training of a visual cortex simulator*. UTFSM, Valparaíso, Chile, 2014.

Appendix A

First Appendix