REDUCE functions provided by redcas

Martin Gregory

August 14, 2024

Contents

1 Introduction

The *redcas* package provides a number of REDUCE procedures which are required for the package to function. While knowledge of these procedures is not needed to use *redcas*, they may be of interest for use in REDUCE itself. There are three types of procedures. The first type produce outputs in a format which can be used by *redcas*. The second type are utility procedures needed by the first type. Finally there are declarations of symbolic procedures for use in algebraic mode using the `symbolic operator` statement. This third type are also utilites used by the first type.

2 Extracting the REDUCE code

All procedures are included in the file *redcas.red* in the *reduce* directory of the installed package. If you are using PSL, *redcas.red* calls *tmprint-psl.red*, also located in the same directory. You can extract the code to a location of your choice using the following code:

```
library(redcas)
file.copy(paste0(redCodeDir(),"/",c("redcas.red","tmprint-psl.red"), "mydir")
```

where *mydir* is a writable directory of your choice. In order that *redcas.red* can find *tmprint-psl.red*, you need to define the environment variable *TMPRINT_PSL_PATH* to point to *tmprint-psl.red* file before starting REDUCE.

3 Output procedures

3.1 asltx: calls arrayltx or exprltx depending on the object type

asltx is a convenience function which calls *arrayltx* 3.2 for arrays, *exprltx* 3.3 for expressions and returns an error for any other type.

Syntax `arrayltx(x, math, mode, name) ;`

Arguments **x** identifier of an object to be typeset.

math string naming a LaTeX math environment in which to enclose each array element. If empty no math environment is written.

mode string naming the print mode: nat|fancy. *nat* is the standard REDUCE output while *fancy* produces LaTeX output using the REDUCE package TMPRINT. If not specified, defaults to nat.

name string providing the name to use when printing expressions.

Details Since REDUCE procedures do not support named arguments[1], all arguments must be specified.

Value None. Called for side effect of producing the output in the desired format.

3.2 arrayltx: converts the flattened list to LaTeX

arrayltx accepts an array of arbitrary dimensions and displays each element using the specified mode.

Syntax `arrayltx (arrx, math, mode) ;`

Arguments **x** identifier of an array.

math string naming a LaTeX math environment in which to enclose each array element. If empty no math environment is written.

mode string naming the print mode: nat|fancy. *nat* is the standard REDUCE output while *fancy* produces LaTeX[output using the REDUCE package TMPRINT. If not specified, defaults to nat.

Details Since REDUCE procedures do not support named arguments, all arguments must be specified.

Value None. Called for side effect of producing the output in the desired format.

3.3 exprltx: converts an expression to LaTeX

Syntax

Arguments **x**

math string naming a LaTeX math environment in which to enclose each array element. If empty no math environment is written.

mode string naming the print mode: nat|fancy. *nat* is the standard REDUCE output while *fancy* produces LaTeXoutput using the REDUCE package TMPRINT. If not specified, defaults to nat.

Details Since REDUCE procedures do not support named arguments, all arguments must be specified.

Value None. Called for side effect of producing the output in the desired format.

[1] lists can be used

4 Utility procedures

4.1 array2flatls: converts an array to a flattened list

array2flatls converts an array to a flattened list using the procedure *array_ to_ list* from the REDUCE package ASSIST and a for loop using the *join* action. This is called by *arrayltx* to allow handling arrays with an arbitrary number of dimensions.

Syntax `array2flatls(arrx) ;`

Arguments **arrx** identifier of array to convert

Value A list containing the elements of the array. The order of the list is unknown to me at the moment.

4.2 asltx_marker: mark output for extraction

asltx_ marker calls *asltx* and encloses the output between a line `##START label` and `##END label`, so that it can be easily extracted from a REDUCE log.

1. also export redSplitOut or just call it from redExtract?
2. do this in 1.0.1 or wait for next version?
3. consider this for the markers:

```
##< label
output
##> label
```

Syntax `asltx_marker (thing, math, mode, name, label) ;`

Arguments **x** identifier of an object to be typeset.

math string naming a LaTeX math environment in which to enclose each array element. If empty no math environment is written.

mode string naming the print mode: nat|fancy. *nat* is the standard REDUCE output while *fancy* produces LaTeX output using the REDUCE package TMPRINT. If not specified, defaults to nat.

name string providing the name to use when printing expressions.

label an arbitrary string to identify the output.

Details This procedure allows extraction of specific outputs from the log of a reduce program which has been run either independently of *redcas* or using or *redcas::redBatch*. While extraction can be done using any program, *redcas::redExtract* provides a way to do this.

Value None. Called for side effect of producing the output in the desired format enclosed in the start and end markers.

4.3 itoa: converts an integer to a string

itoa converts an integer to a string. Useful for formatting output.

Syntax `itoa(integer) ;`

Arguments **integer** an arbitrary integer.

Value a string representation of the integer.

4.4 lisp_dialect

lisp_dialect determines whether REDUCE is running under CSL or PSL

Syntax `lisp_dialect() ;`

Arguments None

Details this function is intended for use in a condition, for example

```
if lisp_dialect = 'csl then ... ;
```

Value the quoted symbol `'csl` or `'psl`

4.5 swget

swget tests whether a REDUCE switch is on or off.

Syntax `swget(s)`

Arguments **s** the identifier of the switch to test

Details if the value of *swget* is *nil* it prints as blank. It should only be used in a condition, for example,

```
if swget(echo) then write "on" else write "off";
```

Value Boolean.

4.6 swtoggle

swtoggle toggles a REDUCE switch.

Syntax `swtoggle(s) ;`

Arguments **s** the identifier of the switch to toggle

Value the new state of the switch.

5 Symbolic procedures

This section describes symbolic procedure which have been declared algebraic by using the `symbolic operator` statement.

5.1 arrayp: array predicate

arrayp is a predicate function to test whether an object is an array or not.

Syntax `array(x)`

Arguments **x** an identifier to test

Details if the value of *arrayp* is *nil* it prints as blank. It should only be used in a condition, for example,

```
if arrayp(x) then ... ;
```

Value Boolean.

5.2 concat: concatenate 2 strings

concat concatenates two strings.

Syntax `result := concat(a, b) ;`

Arguments **a** first string

b second string

Details

Value string containing the concatenation of the first and second strings.

5.3 gettype: distinguishing arrays from "simple" expressions

gettype returns the type of an identifier.

Syntax `gettype(x)`

Arguments **x** the object for which type should be returned

Details *gettype* is used by *asltx* to determine whether to call *arrayltx* or *exprltx*.

Value the type of the object as a quoted identifier, for example, `'array`.

5.4 id2string

id2string returns an identifier's name as a string

Syntax `id_as_string := id2string(x) ;`

Arguments **x** identifier whose name is to be returned as a string

Value string containing the name of the identifier.

5.5 onoff: required by swtoggle

onoff is a symbolic function which sets a switch.

Syntax `onoff(s, bool) ;`

Arguments **s** identifier of the switch.

bool a boolean to set the switch on (t) or off (nil).

Details This is used by swtoggle.

Value None, but check documentation - csl or psl manual?

5.6 tex_string: render string as LaTeX

tex_string prevents REDUCE replacing \ with \textbackslash and {} with \{\} when the FANCY switch is on.

Syntax `result := tex_string(s) ;`

Arguments **s** a string to render as LaTeX

Details *redcas* uses *tex_string* when writing the math environment to ensure the string is not modified.

Value the original string without the unwanted conversions.