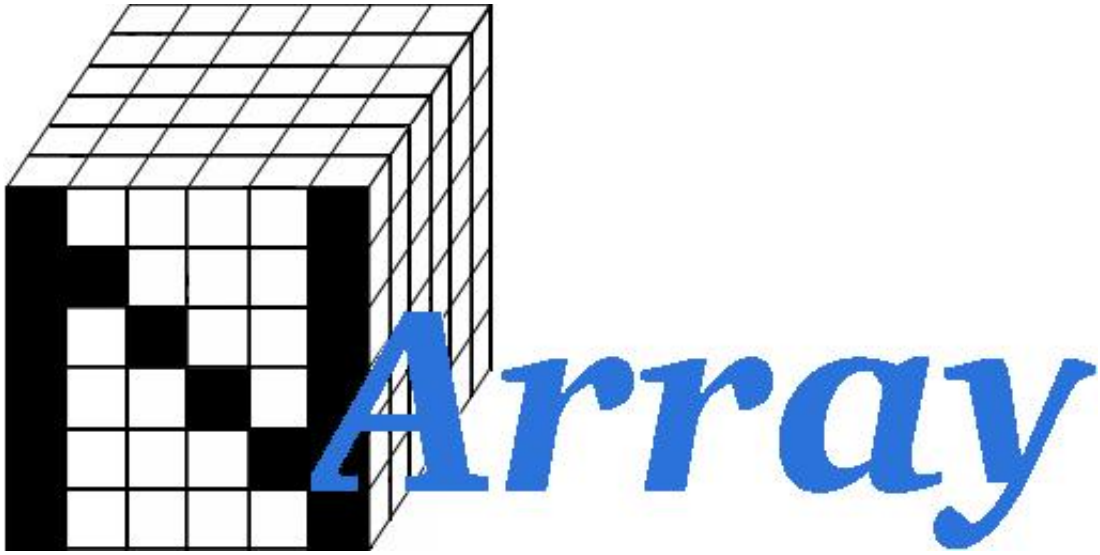

NArray: Manipulating multidimensional data in Tcl

NArray is an extension to help Tcl cope with large in-memory numeric arrays. NArray's require only a few more bytes than the storage required by the array. In addition to providing array referencing and setting, narray allows functions to be mapped over each element of the array. These functions are compiled into byte code for performance about 100x faster than straight tcl and only 5-10x slower than C. (These numbers are ball-park figures, actual results depend on the situation.)

NARRAY LAYOUT AND INDEXES

An narray is stored in memory as a vector of NArrayFloat's (usually float's -- see narray.h). Storage is "C" style, such that an index into an narray index0 ?index1 ...? references the ?indexN + dimN * (indexN-1 + dimN-1 (...index1 + dim1? * index0)...) element. Each indexN can be greater than dimN or even negative, as long as the sum ?indexN + dimN * (indexN-1 + dimN-1 (...index1 + dim1? * index0)...) is within the bounds of the vector. In addition, there may be fewer indexes than dimensions.

NARRAY INVOCATION

The NArray package can be loaded into Tcl or Wish via the native package facility:

```
package require narray
```

NARRAY CREATION

Narray's can be created with the following command:

narray create arrayName dim0 ?dim1 ...?

Creates a command arrayName that represent a multi- dimensional array with dimensions of length dim0 ?dim1 ...?. Returns arrayName.

NARRAY COMMAND

The narray command creates a new Tcl command whose name is the same as the name of the narray. This command may be used to invoke various operations on the narray. It has the following general form:

arrayName option ?arg arg ...?

The following commands are possible for narray's:

arrayName aref index0 ?index1 ...?

Return the element at index0 ?index1 ...?.

arrayName aset index0 ?index1 ...? value

Set the element at index0 ?index1 ...? to value.

arrayName vref variable

Return the value of the variable variable.

arrayName vset variable value

Set the variable variable to value.

arrayName vars

Return a list of variables in arrayName.

arrayName map code ?{array_var0 array0} ...?

Compile code and apply it to each element of arrayName. Make the narray array0 available to code as array_var0.

arrayName dimensions

Return the dimensions of arrayName.

arrayName collapse direction result

Sum over all elements in the indicated direction, saving the result to the narray result.

arrayName export offset size result

Extract a subset of arrayName of the specified size and at the specified offset indices into the narray result.

arrayName import offset source

Import the contents of the narray source into arrayName at the specified offset indices.

arrayName status

Return some status information about arrayName (currently, the kilobytes used by the narray and its debug status.)

arrayName debug level

Set the debug flags of arrayName to level. The debug flags are a bitwise OR of:

1 Print parsing information
2 Trace stack machine execution
4 Print compiled code before executing it

NARRAY LANGUAGE

The arrayName map command compiles and applies a sequence of code to each element in a narray. Statements in the language are seperated by semi-colons or by newlines. The narray language includes standard math (including +=, -=, *=, /= and ?:). = is used for assignment. The following forms may appear on the left hand side of an assigment:

var

The variable var. Variables may contain numeric values only.

[?index0, index1, ...?]

The current element, modified by index0, index1, ... For example, [] is the current element, [-1] is the previous element, and [0,1] is the element in the same row but next column in a 2D matrix.

array_var[?index0, index1, ...?]

Accesses elements of the narray array_var.

The value of variables and array elements may be referenced by prefixing a $ to them. For example, $foo is the value of the variable foo. In addtion @N may be used to reference the value of the N'th index of the current element. For example, @0 is the row number and @1 is the column number (for a 2D matrix.)

Functions may be called as func(?arg0, ...?) Strings enclosed by "e;'s may be passed to functions. The following functions are available:

sin(x)

The sine of x.

cos(x)

The cosine of x.

tan(x)

The tangent of x.

asin(x)

The arc sine of x.

acos(x)

The arc cosine of x.

atan(x)

The arg tangent of x.

sinh(x)

The hyperbolic sine of x.

cosh(x)

The hyperbolic cosine of x.

tanh(x)

The hyperbolic tangent of x.

exp(x)

e ^ x.

log(x)

The natural log of x.

log10(x)

The base 10 log of x.

sqrt(x)

The square root of x.

ceil(x)

The smallest integral value not less than x.

floor(x)

The largest integral value not greater than x.

atan2(x, y)

The arc tangent function of two variables.

pow(x, y)

x raised to the y power.

fmod(x, y)

The floating-point remainder of x / y.

printf(format_string, ?arg?, ...)

A limited form of printf that understands the d, f, g, and s formats.

fprintf(file_number, format_string, ?arg?, ...)

Like printf, but output goes to the file handle identified by file_number. Tcl's file handles are of the form filefile_number.

tcl_eval(string, ?arg?, ...)

Concatenate string and any arguments (numeric arguments are converted to strings and preceded by a space) and evaluate in Tcl in the current context.

VARIABLES

The following variables are set by this extension:

narray_library

The location of the Tcl library. This can be over- ridden by the environment variable NARRAY_LIBRARY.

narray_version

The version number of the narray extension, in the form major.minor.

LIBRARY COMMANDS

The file narray.tcl contains these useful library routines:

pnarray arrayName

Print the narray arrayName.

narray_delete arrayName

Delete the narray arrayName. This is the same as rename arrayName {}.

Here's an example:

```
% package require narray        ;# load narray package
0.81
% narray create cube 64 64 64   ;# cube is an 64x64x64 float array
cube
% cube status                   ;# 64x64x64 * sizeof(float) = 1MB
 1024.12KB used, debug 0
% cube aref 0 0 0               ;# return the element (0,0,0)
0
% cube aset 0 0 0 10            ;# set (0,0,0) to 10
10
% cube map { [] += 5; }         ;# add 5 to each element
% cube aref 0 0 0               ;# (0,0,0) is now 15
15
% cube vset sum 0               ;# set the variable sum to 0
0
% cube map { sum += $[]; }      ;# sum the elements
% cube vref sum                 ;# get the value of the variable sum
1.31073e+06                     ;# the sum of the elements is 1310730
% expr 64*64*64*5+10            ;# just checking...
1310730
```

OBTAINING NARRAY

The narray extension can be obtained from the Neosoft archives or via anonymous ftp from our site.

HISTORY

The narray extension was originally written circa 1994 by Sam Shen at Lawrence Berkeley Laboratories. It was then adopted by Nick Maliszewskyj <nickm@nist.gov> and Przemek Klosowski <przemek@nist.gov> at the National Institute of Standards and Technology. Please contact Nick or Przemek if you have questions, modifications, bug reports, or fixes.

Last modified 1-Sep-1998.