CoCoA-5 - Feature #1296

Matrixrow-functions

16 Jun 2019 21:54 - Julian Danner

Status:	In Progress	**Start date:**	16 Jun 2019
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	10%
Category:	enhancing/improving	**Estimated time:**	0.00 hour
Target version:	CoCoA-5.?.?	**Spent time:**	0.85 hour

Description

I ran into a problem concerning matrix-rows. Namely, I wanted to implement a function returning the Hamming-weight of a matrixrow (and/or vector,list,moduleelem,...). However, it turned out that it is not easy to even determine the number of columns of a given MATRIXROW without access to its corresponding matrix, for which we could just use NumCols. Also len and a cast to LIST do not work.

So, is there any way to *simply* find the length of a given MATRIXROW without accessing its matrix?
(One possibility I can think of is to run over all entries until an invalid-column-index error is thrown, but that seems to be a pretty ugly workaround...)

Related issues:

Related to CoCoA-5 - Feature #487: ScalarProduct accepts MatrixRow?	**New**	**21 Mar 2014**
Related to CoCoA-5 - Slug #1597: GetRow/GetRows are extraordinarily slow	**Closed**	**27 May 2021**

History

#1 - 17 Jun 2019 11:43 - John Abbott

- Category set to enhancing/improving

- Target version set to CoCoA-5.?.?

The "easy solution" is to use GetRow(M,1) or R:=GetRows(M); R[1] instead of M[1]. But this makes copies of the matrix entries, so will surely be slow for large matrices (or matrices with large entries).

As I recall MATRIXROW was created largely to support the old CoCoA-4 syntax for accessing matrix entries: namely M[1][2] was an alternative to M[1,2]. It may have allowed a slightly neater implementation of gaussian reduction... I believe a command such as M[1] := M[1]+M[2]; worked as expected, but in CoCoA-5 it is not permitted.

Currently, not many operations are permitted on a MATRIXROW.
If we do allow more, we should also ensure that CoCoALib allows
similar operations.

Note that MATRIXCOL does not exist.

#2 - 17 Jun 2019 11:46 - John Abbott

- Status changed from New to In Progress

- % Done changed from 0 to 10

The specific request to make len or NumCols work for a MATRIXROW should not be too hard to achieve. Which function name? I suppose NumCols is more precise...

#3 - 18 Jun 2019 15:15 - Anna Maria Bigatti

From what you say, I think you are passing a MATRIXROW as an argument (because you say you cannot call NumCols).

I have two suggestions for you:
1 - pass the MATRIX and the INT index (so you can use **NumCols**) - no copies
2 - pass the LIST GetRow(M,n) (so you can use **len**) - makes copies

John Abbott wrote:

> As I recall MATRIXROW was created largely to support the old CoCoA-4 syntax for accessing matrix entries: namely M[1][2] was an alternative to M[1,2].

I confirm this: MATRIXROW is just a matrix + an index.
It behaves like a pointer, and this makes it very dodgy/dangerous/fragile, with no reference counting :scream: !!

Conclusion: I suggest limiting even further (!!) this dangerous type, so that we don't induce into temptation ;-)
In particular, we should prohibit passing MATRIXROW as function argument, because it behaves differently from other types (by ref, insteaf of by value).

#4 - 18 Jun 2019 15:17 - Anna Maria Bigatti

- Related to Feature #487: ScalarProduct accepts MatrixRow? added

#5 - 27 May 2021 12:06 - John Abbott

- Related to Slug #1597: GetRow/GetRows are extraordinarily slow added