Chandra X-ray Center

Bugs: dmtcalc

Bugs

1. ***When using status bits in an expression, only the last condition is checked.***

   When using status bits in an expression such "`status==X1F,X3T,X18F`", only the last condition (here, "`X18F`") is checked.

   Workaround:

   Set the logic explicitly:

   - logical "AND": `status=((bits==X1T)&&(bits==X2T))`
   - logical "OR": `status=((bits==X1T)||(bits==X2T))`

2. ***The expression "if(a)then(b)" does not work as expected in `dmtcalc` if two different columns are used in the comparison.***

   For example,

   ```
   unix% dmtcalc in.fits out.fits "expr=if(pi>500)then(pi=-1)"
   ```
   works as expected ("pi" used for both pieces of the conditional), but

   ```
   unix% dmtcalc in.fits out.fits "expr=if(energy>5000)then(pi=-1)"
   ```
   does not work.

3. ***The tool does not follow the usual order of operations of mathematics when evaluating an expression.***

   For example, the expression

   ```
   x+814*24*3600-260086780.04
   ```
   is treated as

   ```
   x+(814*24*(3600-260086780.04))
   ```
   instead of the expected

   ```
   x+(814*24*3600)-260086780.04
   ```
   Workaround:

   Include the parentheses, as shown in the last code snippet, to force the correct order of operations.

4. ***When creating a new double vector column, the results are incorrect if one (but not both) of the values in the expression are integers, even if it is cast as a double.***

   For example, this file will work:

```
unix% cat ok.expr
(r=(double){3.5,4.})

unix% dmlist output_ok.fits'[cols r]' data
--------------------------------------------
Data for Table Block HISTOGRAM
--------------------------------------------

ROW R[2]

    1 [ 3.50 4.0]
    2 [ 3.50 4.0]
    3 [ 3.50 4.0]
(etc.)
```

while this does not:

```
unix% cat not_ok.expr
(r=(double){3.5,4})

unix% dmlist output_not_ok.fits'[cols r]' data
--------------------------------------------
Data for Table Block HISTOGRAM
--------------------------------------------

ROW R[2]

    1 [ 3.50 0]
    2 [ 3.50 0]
    3 [ 3.50 0]
(etc.)
```

5. ***Virtual columns cannot be used in an expression***

   For example, when using an event list, one cannot say

   ```
   unix%  dmtcalc evt1.fits foo.fits exp='dra=(ra-ra_nom)'
   ```
   because the 'ra' column is not a real column in the file; see the virtual columns dictionary entry for an explanation.

   Workaround:

   Use Data Model column renaming:

   ```
   unix%  dmtcalc evt1.fits"[cols foo=ra,*]" foo.fits expr='dra=(foo-ra_nom)'
   ```
6. ***Units are not preserved, even for simple calculations.***

   If you calculate npha as npha=pha; which should have units of PHA [adu]; but it will be unitless. Even more confusing is pha=pha will be unitless since pha in the output file is treated a new column.
7. ***The speed basically scales as O(N^2): small files are not so bad, but big files slow down exponentially.***
8. ***The #nan can be use for float–point NaN checks, but integer NULL values cannot be checked the same way.***
9. ***Problems accessing individual elements in a vector and/or array column***

   There are various problems doing things like:

   ```
   pha[9]=1;
   foo=pha[9];
   ```
   where you try to access the individual elements in a vector and/or array column.

Workaround:

10. ***Changing the value of an existing column***

   There are issues when changing values of existing columns: what info is kept, what is thrown away (e.g. data types, null, units, descriptions, order in file). In particular there are various problems with arrays vs. vector vs. vector–array columns.

11. ***There is no way to access elements of an array column such as `PHAS`, which is a 3x3 array (or 5x5 in VFAINT mode).***

12. ***Booleans do not work on array columns***

   For example, you ***cannot*** do `above=(phas>20)` since the `PHAS` array contains 9 (or 25) values. Currently only the first value in the array is checked.

13. ***Using vector components in a complicated expression may cause `dmtcalc` to hang.*** *(01 Jun 2006)*

   For example:

```
expression="dist=((4125-x)^2)-((4025-y)^2)"
```
   Workaround:

   Refer to the component by vector notation instead:

```
expression="dist=((4125-sky[0])^2)-((4025-sky[1])^2)"
```
   `dmtcalc` tries to do this subsitution internally in the first example, but fails.

Workaround: