To: MTB Distribution
From: NSDavids
Date: June 30, 1980
Subject: A Performance Comparison Between MR6.5 and MR8.0 MRDS

I PURPOSE

The purpose of this MTB is to show performance differences between the old MR6.5 MRDS and the new MR8.0 MRDS.

II CAVEAT

The performance testing for the MR8.0 PSP MRDS has not been exhaustive, but has provided a foundation for ongoing performance testing and the development of a comprehensive performance testing benchmark. The tests performed do provide an indication of the performance relationship of MRDS 6.5 and MRDS 8.0 PSP for simple 1-relation requests. The extrapolation of these results to more complex requests is of questionable validity. Changes to MRDS are being made which may affect performance and hence the results presented here.

III DESCRIPTION OF TESTS

All tests were performed against the data bases rfxbs and c.

```
rfxbs:
   domain:
            kh          fixed bin (10),
            kt          fixed bin (17),
            indx        fixed bin (11),
            data        fixed bin (35);
   relation:
            rel1        (kh* kt* indx data),
            rel2        (kh* kt* indx data),
            rel3        (kh* kt* indx data);
   index:
            rel1        (indx),
```

```
                rel2        (indx),
                rel3        (indx);
        c:
            domain:
                kh          character (10),
                kt          character (17),
                indx        character (12),
                data        character (35);
            relation:
                rel1        (kh* kt* indx data),
                rel2        (kh* kt* indx data),
                rel3        (kh* kt* indx data);
            index:
                rel1        (indx),
                rel2        (indx),
                rel3        (indx);
```

For each data model two sets of data called unq and dup were used. In the unq data set all attributes in a tuple have the same value which ranges from 1 to 1000. In the dup data set all attributes in a tuple except for the indx attribute have the same value which ranges from 1 to 1000; the indx attribute has a value of 6 in all 1000 tuples.

A. STORE TESTS

In doing a store there are two possible ways to communicate the values to be stored to MRDS; all the values may be contained in a structure and the structure passed to MRDS or the individual variables may be passed to MRDS. Also while the first store must name the target relation all subsequent stores may either name the relation or use the "-another" construct. All four combinations are executed. Timing data is gathered for the first store, the middle 998 stores and the last store.

Note that the declaration of the MRDS attributes and the corresponding variables or elements in the structure are the same so that MRDS does not have to do any conversion.

B. RETRIEVE, DELETE, and MODIFY TESTS

The selection expressions used in the retrieve, delete and modify tests only span 1 relation. There are three sets of tests which select the first 10 tuples (keys 1 - 10), the middle 10 tuples (keys 496 - 505) and the last 10 tuples

(keys 991 - 1000). Within each test set selection is made by comparing a data attribute (data), an indexed attribute (indx), a key head attribute (kh), a key tail attribute (kt) and a key (kh, kt) to a constant value. There is also one test which does not have a where clause and therefore selects all 1000 tuples. For the retrieve operation timing data is gathered on the first retrieve, all subsequent retrieves that return data, and on the last retrieve which returns the error code tuple_not_found. For the delete and modify operations there is no way to separate actions on individual tuples so only 1 datum is collected.

Note that for the retrieve tests all the attributes of a tuple are returned into individual variables (as opposed to a structure) and that those variables have the same declaration as the corresponding MRDS attributes so that no conversion (by MRDS) is necessary. Also in the modify tests, tests are run which select only the data attribute, only the indx attribute, and both the indx and data attribute. As in the retrieve tests no structure breaking or conversion is necessary.

IV RESULTS

The results are expressed as a percentage improvement of new MRDS to old MRDS and are calculated by the formula:

((old_time - new_time) / old_time) * 100

A. OVERALL PERFORMANCE DIFFERENCE

By using the results from all tests without differentiating by operation, data type, or data set the overall performance improvement of new MRDS is 31.294 %.

B. OVERALL PERFORMANCE DIFFERENCE BY OPERATION

The improvement of an operation is calculated by not differentiating results from different data types or data sets.

For the store operation: 6.907 %
For the retrieve operation: 74.861 %
For the modify operation: 8.465 %
For the delete operation: 11.405 %

C. BY OPERATION AND DATA TYPE

By differentiating by operation and data type the results may be further broken down into:

store operation rfxbs data base: 3.405 %
c data base: 5.625 %

retrieve operation on the rfxbs data base: 26.953 %
c data base: 85.624 %

modify operation on the rfxbs data base: 8.465 %

delete operation on the rfxbs data base: 15.680 %
c data base: 5.857 %

Note: A bug in old MRDS prevented the execution of modify tests with character data types.

V CONCLUSIONS

Significant improvement in the performance of some of the MRDS operations (i.e. retrieve) has been made, in other areas (store) more work is needed. Analysis of the results at a finer level of detail than presented here has indicated several possible performance bugs.

These tests represent the start of a comprehensive performance benchmark, a set of tests that will not only tell us how fast MRDS will do something but how sensitive that time is with respect to the many variables that determine the structure of a data base. Over 1135 minutes of CPU time was used in executing these tests, the machine resources needed for a comprehensive set of sets will be considerably greater.