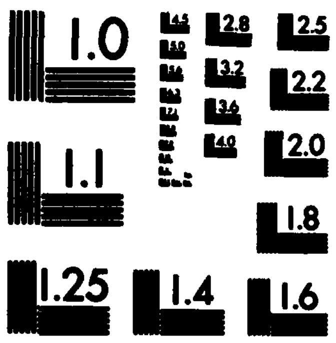
AD-A148 191 BENCHMARKING DATABASE SYSTEMS IN MULTIPLE BACKEND CONFIGURATIONS(U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA S A DEMURJIAN ET AL. NOV 84 NPS52-84-020 1/1

UNCLASSIFIED F/G 9/2 NL

END

FILMED

DTIC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0

(2)

NPS52-84-020

NAVAL POSTGRADUATE SCHOOL

Monterey, California

BENCHMARKING DATABASE SYSTEMS IN MULTIPLE BACKEND CONFIGURATIONS

Steven A. Demurjian and David K. Hsiao

November 1984

Prepared for: Chief of Naval Research
Arlington, VA 22217

NAVAL POSTGRADUATE SCHOOL
Monterey, California

Commodore R. H. Shumaker
Superintendent

D. A. Schrady
Provost

The work reported herein was supported in part by the Foundation Research Program of the Naval Postgraduate School with funds provided by the Chief of Naval Research.

This report was prepared by:

[illegible]
DAVID K. HSIAO
Professor of Computer Science

Reviewed by:

[illegible]
BRUCE J. MAC LENNAN
Acting Chairman
Department of Computer Science

Released by:

[illegible]
KNEALE T. MARSHALL
Dean of Information and
Policy Sciences

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NPS52-84-020	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Benchmarking Database Systems in Multiple Backend Configurations		5. TYPE OF REPORT & PERIOD COVERED
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Steven A. Demurjian: David K. Hsiao		8. CONTRACT OR GRANT NUMBER(s) N00014-84 WR-24058
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 61152N: RR000-01-10 N000149 WR41001
11. CONTROLLING OFFICE NAME AND ADDRESS Chief of Naval Research Arlington, Virginia 22217		12. REPORT DATE November 1984
		13. NUMBER OF PAGES 9
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report)		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number)		

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The aim of this performance evaluation is twofold: (1) to devise benchmarking strategies for and apply benchmarking methodologies to the measurement of a prototyped database system in multiple backend configurations, and (2) to verify the performance claims as projected or predicted by the designer and implementor of the multi-backend database system known as MBDS.

Despite the limitation of the backend hardware, the benchmarking experiments have proceeded well, producing startling results and good insights. By

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

collecting maroscopic data such as the response time of the request, the external performance measurements of MBDS have been conducted. The performance evaluation studies verify that (a) when the database remains the same the response time of a request can be reduced to nearly half, if the number of backends and their disks is doubled; (b) when the response set of a request doubles, the response time of the query remains nearly constant, if the number of backends and their disks is doubled. These were the performance claims of MBDS as predicted by its designer and implementor.

BENCHMARKING DATABASE SYSTEMS IN MULTIPLE BACKEND CONFIGURATIONS *

Steven A. Demurjian
Department of Computer Science
Naval Postgraduate School
Monterey, CA 93943
(408)-646-3391

and

David K. Hsiao
Department of Computer Science
Naval Postgraduate School
Monterey, CA 93943
(408)-646-2253

November 1984

ABSTRACT

The aim of this performance evaluation is twofold: (1) to devise benchmarking strategies for and apply benchmarking methodologies to the measurement of a prototyped database system in multiple backend configurations, and (2) to verify the performance claims as projected or predicted by the designer and implementor of the multi-backend database system known as MBDS.

Despite the limitation of the backend hardware, the benchmarking experiments have proceeded well, producing startling results and good insights. By collecting macroscopic data such as the response time of the request, the external performance measurements of MBDS have been conducted. The performance evaluation studies verify that (a) when the database remains the same the response time of a request can be reduced to nearly half, if the number of backends and their disks is doubled; (b) when the response set of a request doubles, the response time of the query remains nearly constant, if the number of backends and their disks is doubled. These were the performance claims of MBDS as predicted by its designer and implementor.

---

* The work reported herein is supported by Contract N00014-84-WR-24058 from the Office of Naval Research and conducted at the Laboratory for Database Systems Research, Naval Postgraduate School, Monterey, CA 93943.

1. INTRODUCTION

The multi-backend database system (MBDS) is a database system designed specifically for capacity growth and performance enhancement. MBDS consists of two or more minicomputers and their dedicated disk systems. One of the minicomputers serves as a controller to broadcast the requests to and receive the results from the other minicomputers, which are configured in a parallel manner and are termed as backends. All the backend minicomputers are identical, and run identical software. The database is evenly distributed across the disk drives of each backend by way of a cluster-based data placement algorithm unknown to the user. User access to the MBDS is accomplished either via a host computer, which in turn communicates with the MBDS controller, or with the MBDS controller directly. Communication between the controller and backends is accomplished using a broadcast bus. An overview of the system architecture is given in Figure 1.

There are two basic performance claims of the multi-backend database system, which have been projected in the original design goals [Hsia81a, Hsia81b]. The first claim states that if the database size remains constant, then the response time of requests processed by the system is inversely proportional to the multiplicity of backends. This claim implies that by increasing the number of backends in the system and by replicating the system software on the new backends, MBDS can achieve a reciprocal decrease in the response time for the same requests. The second claim states that the response time of requests is invariant when the response set and the multiplicity of backends increase in the same proportion. This claim implies that when the database size grows, the response set for the same requests will grow. By increasing the number of backends accordingly, MBDS can maintain a constant response time.

In this paper we provide a preliminary evaluation of the validity of the MBDS performance claims. The main focus of this paper is on the external performance measurement of MBDS. The external performance measurement evaluates a system by collecting the response times of requests. External performance measurement is a macroscopic evaluation of the system. Ingres, Oracle, and the Britton-Lee IDM/500, have all been evaluated using external performance measurement techniques [Stra84, Schi84].

The remainder of this paper is organized as follows. In Section 2 we provide a brief overview of the multi-backend database system. In Section 3 we discuss the general testing strategy that was used to evaluate the system. In Section 4 we examine the evaluation results. Finally, in Section 5 we conclude this paper and summarize the results.

2. THE MULTI-BACKEND DATABASE SYSTEM (MBDS)

The current hardware configuration of MBDS consists of a VAX-11/780 (VMS OS) running as the controller and two PDP-11/44s (RSX-11M OS) running as backends. Intercomputer communication is supported by three parallel communication links (PCL-11Bs), which is a time-divisioned-multiplexed bus. An overview of MBDS can be found in [Kerr82]. The implementation efforts are documented in [He82, Boyn83b, Demu84]. MBDS is a message-oriented system (see [Boyn83a]). In a message-oriented system, each process corresponds to one system function. These processes, then, communicate among themselves by passing messages. User requests are passed between processes as messages. The message paths between processes are fixed for the system. The MBDS processes are created at the start-up time and exist throughout the entire running time of the system.

MBDS provides a centralized database system where the database itself is evenly distributed across the backend processors. Only a single copy of the database is stored. The underlying data model for MBDS is the attribute-based data model [Hsia70]. The attribute-based data model stores data in files of records. MBDS stores records of a file in clusters. A *cluster* is a group of records such that every record in the cluster satisfies the same set of attribute-value pairs or ranges. Thus, a file is divided into one or more clusters. The distribution of the database is accomplished using a cluster-based data placement algorithm.

The cluster-based data placement algorithm is arbitrated and managed by the controller. When a new cluster is defined, the backend processor notifies the controller. The controller then decides which backend will insert the new record. Under the direction of the controller, the chosen backend will continue to insert records of the new cluster, until the backend processor fills a block of secondary memory storage. When this occurs, the backend processor notifies the controller that the block is full. The controller then directs another backend for the insertion of new records of the cluster. In a multiple-backend configuration, the controller attempts to achieve a block-parallel-and-record-serial operation for any subsequent access to the records of the cluster.

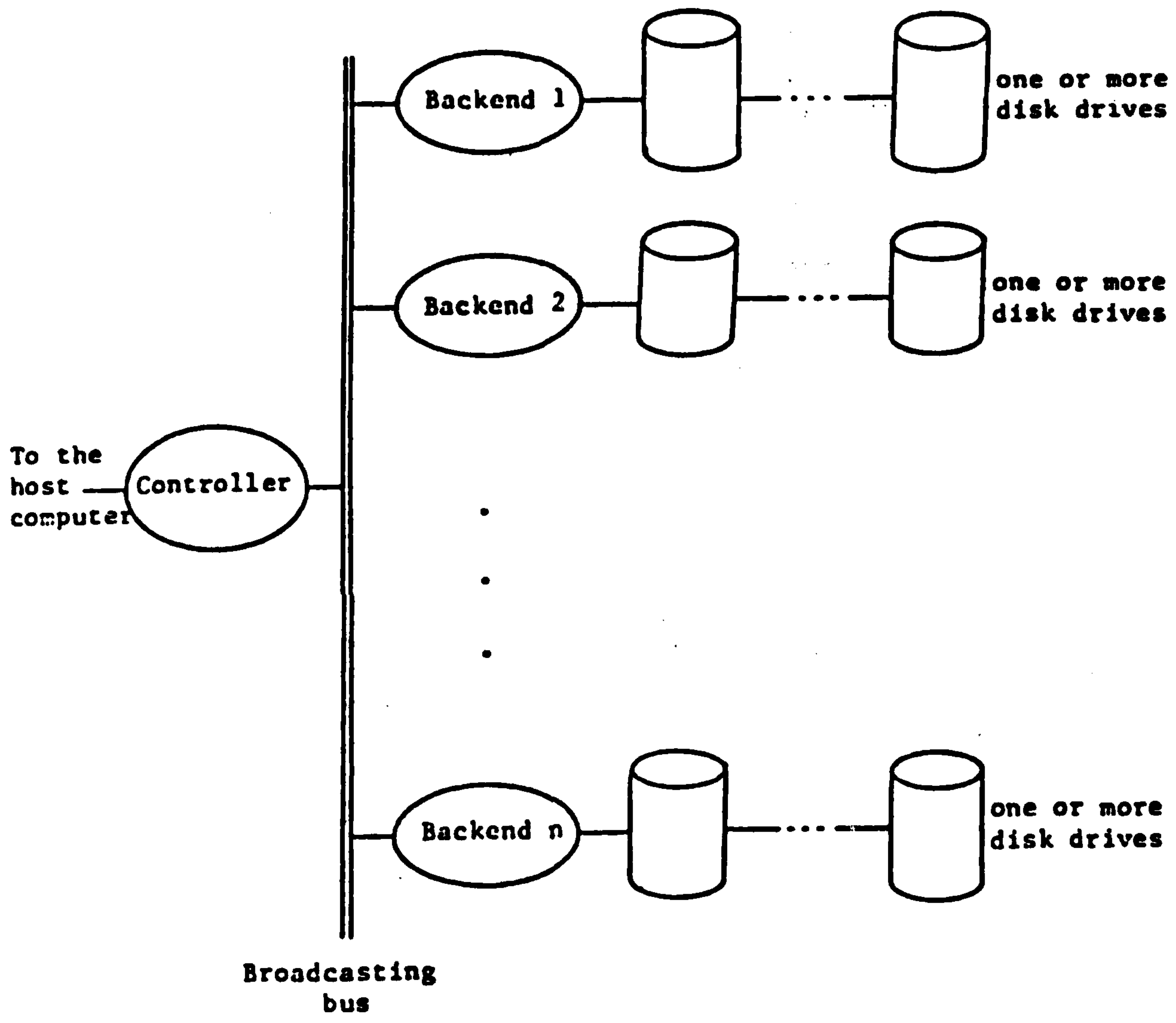


Figure 1. The MDBS Hardware Organization

Let's trace through an example. Suppose that our system has four backend processors, the average size of a record is 200 bytes, and the size of a block of secondary storage memory is 4K (so, each block contains approximately 20 records). A new cluster of 100 records, say C, is defined. The controller picks say, Backend 3, for inserting records of cluster C. Backend 3 will insert 20 records into a block for the cluster C under the direction of the controller. Then the controller will have Backend 4 insert records of cluster C. After Backend 4 has inserted 20 records, the controller will cycle to Backend 1, and continue the round-robin process until all 100 records are placed on the secondary storage blocks. For the next new cluster, say, C', the controller will then pick Backend 4, since Backend 3 is the last backend used by the previous cluster in the algorithm.

3. THE BENCHMARK STRATEGY

In this section we analyse the basic benchmark strategy for the preliminary performance evaluation of the multi-backend database system. The benchmark strategy focuses on collecting macroscopic measurements on the systems performance. Macroscopic measurements correspond to the external performance measurement of the system, which collects the response time of requests that are processed by the system. To adequately conduct the external performance measurement of the system, software was developed to collect timing information and data. The performance software was bracketed in conditional compilation statements to facilitate an easy transition between a testing system and a running system.

The rest of this section is organized as follows. First, we give a high-level description of the test database organization and system configurations used in the performance evaluation. Next, we examine the request set used to collect the timings. Finally, we review the relevant tests that are to be conducted, and the measurement statistics that are collected and calculated.

3.1. The Test Database Organization and Testing Configurations

The test database was constructed using a record size of 200 bytes. A total of 24 clusters are defined for the test database. The virtual and physical memory limitations of each backend restricted the database size to a maximum of 1000 records per backend. This limitation, coupled with the need to verify the two performance claims, led us to the specification of three different system configurations for the MBDS performance measurements. Table 1 displays the configurations.

Test A configures MBDS with one backend and one thousand records in the test database. Test B configures MBDS with two backends and one thousand records split evenly between the backends. The transition *from* Test A to Test B is used to verify the first performance claim (see Section 1). Tests A and B have 23 clusters that contain 40 records and one cluster that contains 80 records. In Test A, all of the records are stored on the single backend. In Test B, each backend stores 20 records for the first 23 clusters and 40 records for the last cluster.

Test C also configures MBDS with two backends, but, the size of the database is doubled to two thousand records. The transition from Test A to Test C is used to verify the second performance claim (see Section 1). Test C has 23 clusters that contain 80 records each and one cluster that contains 160 records. In Test C, each backend stores 40 records for each of the first 23 clusters and 80 records for the last cluster. Notice that the record totals per cluster per backend are the same for Test A and Test C.

3.2. The Request Set

In this section we review the retrieve requests that are used to benchmark MBDS. The retrievals, shown in Table 2, are a mix of single and double predicates. There are two directory attributes and thirty-one non-directory attributes in each record. The directory attributes, INTE1 and INTE2, are

TEST	No. of Backends	Records/Backend	Database Size
A	1	1000	200K bytes
B	2	500	200K bytes
C	2	1000	400K bytes

Table 1. The Measurement Configurations

integer-valued, and are used for the cluster definition and formation. INTE1 is defined using 5 attribute-value ranges, while INTE2 is defined using 24 attribute-value ranges. The non-directory attributes are used as fillers for the 200-byte record. The retrieve requests given in Table 2 are specified using equality and inequality predicates, to control the search space when accessing the database records.

In Table 3 we present a high-level analysis of the request set given in Table 2. We focus on specifying two characteristics for each retrieve request in the request set; the number of clusters examined by the particular retrieve request and the volume of the database information that is retrieved. The values in Table 3 apply to the three testing configurations, A, B, and C, with one exception. The numbers in parenthesis in the third column represent the number of records retrieved for Test C.

3.3. The Measurement Strategy, Statistics and Limitations

The basic measurement statistics used in the performance evaluation of MBDS is the response time of request(s) that are processed by the database system. The *response time* of a request is the time between the initial issuance of the request by the user and the final receipt of the entire request set for the request. The response times are collected for the request set (see Table 2) for each of the three configurations (see Table 1). Each request is sent a total of ten times per database configuration. The response time of each request is recorded. We determine that ten repetitions of each request produce an acceptable standard deviation. Upon completion of the ten repetitions for a request, we calculate the mean and the standard deviation of the ten response times. There are two main statistics that we calculate to evaluate the MBDS performance claims, the response-time improvement and the response-time reduction.

The *response-time improvement* is defined to be the percentage improvement in the response time of a request, when the request is executed in n backends as opposed to one backend and the number of

Request Number	Retrieval Request
1	(INTE1 = 10) or (INTE1 = 230)
2	(INTE2 =< 250)
3	(INTE2 =< 500)
4	(INTE1 =< 1000)
5	(INTE1 =< 200) or (INTE1 >= 801)
6	(INTE1 =< 400) or (INTE1 >= 601)
7	(INTE1 <= 201)
8	(INTE1 <= 401)
9	(INTE1 <= 201) or (INTE1 >= 800)

Table 2. The Retrieval Requests

Request Number	Number of Clusters Examined	Volume of Database Retrieved
1	2	2(4) records
2	7	25%
3	13	50%
4	24	100%
5	9	40%
6	19	80%
7	10	20% + 1(2) record
8	15	40% + 1(2) record
9	19	40% + 2(4) records

Table 3. The Number of Clusters Examined and the Percent of the Database Retrieved

records in the database remains the same. Equation 1 provides the formula used to calculate the response-time improvement for a particular request, where Configuration Y represents n backends and Configuration X represents one backend. The response-time improvement is calculated for the configuration pair (A, B). The configuration pair (A, B) is evaluated for the retrieve requests (1) through (9) (see Table 2).

$$\begin{matrix}\textit{The}\\ \textit{Response Time}\\ \textit{Improvement}\end{matrix} = 100\% * \left[1 - \left(\frac{\begin{matrix}\textit{The Response}\\ \textit{Time of}\\ \textit{Configuration } X\end{matrix}}{\begin{matrix}\textit{The Response}\\ \textit{Time of}\\ \textit{Configuration } Y\end{matrix}} \right) \right]$$

Equation 1. The Response-Time-Improvement Calculation

The *response-time reduction* is defined to be the reduction in response time of a request, when the request is executed in n backends containing nx number of records as opposed to one backend with x number of records. Equation 2 provides the formula used to calculate the the response-time reduction for a particular retrieval request, where configuration X represents one backend with x records and configuration Z represents n backends, each with x records. The response-time reduction is calculated for the configuration pair (A, C), for the retrieve requests (1) through (9).

$$\begin{matrix}\textit{The}\\ \textit{Response Time}\\ \textit{Reduction}\end{matrix} = 100\% * \left[1 - \left(\frac{\begin{matrix}\textit{The Response}\\ \textit{Time of}\\ \textit{Configuration } Z\end{matrix}}{\begin{matrix}\textit{The Response}\\ \textit{Time of}\\ \textit{Configuration } X\end{matrix}} \right) \right].$$

Equation 2. The Response-Time-Reduction Calculation

Finally, we examine the limitations of the testing strategy. The last two versions of MBDS differ in the implementation of the directory tables. The newest version of the system, called Version F, implements the directory tables on the secondary storage. The previous version, called Version E, stored the directory tables in the primary memory. The major roadblock that we have encountered in the performance measurement of MBDS has been the hardware limitations of the backend processors (PDP-11/44). With only 64K of virtual memory per process and a total of 256K physical memory, we found that we could not increase the MBDS system parameters to permit an extensive test of the system on a large database. These restrictions have forced us to benchmark the primary-memory-based directory management version of the system which, excluding the directory table management routines, is nevertheless equivalent in functionality to Version F.

4. THE BENCHMARKING RESULTS

In this section, we present the results obtained from the performance measurement of MBDS. In particular, we review the results of external performance measurement, in the hope of verifying the MBDS performance and capacity claims. One final note, the units of measurement presented in the tables of this section are expressed in seconds.

Table 4 provides the results of the external performance measurement of MBDS. There are three parts to Table 4. Each part contains the mean and the standard deviation of the response times for requests (1) through (9), which are outlined in Section 3.2. The three parts of Table 4 represent three different configurations of the MBDS hardware and the database capacity. The first part has configured MBDS with one backend and the database with 1000 records on its disk. The second part has configured MBDS with two backends, with the database of 1000 records, split evenly between the disks of the backends. The third part has configured MBDS with two backends and with a database doubled of 2000 records, where the disk of each backend has 100 records.

Request Number	One Backend 1K Records (A)		Two Backends 1K Records (B)		Two Backends 2K Records (C)	
	mean	stdev	mean	stdev	mean	stdev
1	3.208	0.0189	2.051	0.0324	3.352	0.0282
2	13.691	0.0255	7.511	0.0339	14.243	0.0185
3	26.492	0.0244	14.164	0.0269	26.737	0.0405
4	52.005	0.0539	26.586	0.0294	52.173	0.0338
5	21.449	0.0336	11.309	0.0375	21.550	0.0237
6	42.235	0.0326	21.622	0.0424	42.287	0.0400
7	12.285	0.0408	6.642	0.0289	12.347	0.0371
8	22.532	0.0296	11.764	0.0300	22.583	0.0110
9	23.913	0.1115	12.624	0.0350	24.169	0.0181

Table 4. The Response Time Results

Given the data presented in Table 4, we can now attempt to verify or disprove the two MBDS performance claims. We begin by calculating the response-time improvement for the nine requests. In Table 5 we present the response-time improvement for the data given in Table 4. Notice that the response-time improvement is lowest for request (1), which represents a retrieval of two records of the database. On the other hand, the response-time improvement of request (4), which retrieves all of the database information is highest, approaching the upper bound of fifty percent. In general, we find that the response-time improvement increases as the number of records retrieved increases. This seems to support a hypothesis that even if the response set (therefore the database) is larger, the response-time improvement will remain at a relatively high level (between 40 an 50 percent).

Request Number	Response-Time Improvement (A,B)
1	36.07
2	45.14
3	46.53
4	48.94
5	47.27
6	48.81
7	45.93
8	47.79
9	47.21

Table 5. The Response-Time Improvement Between Configurations A and B.

Next, we calculate the response-time reduction for each of the nine requests. In Table 6 we present the response-time reductions for the data given in Table 4. Notice that the response-time reduction is worst for request (1), which represents a retrieval of two records of the database. On the other hand, the response-time reductions for the requests which access larger portions of the database, requests (4) and (6), have only a small response-time reduction. In general, we found that the response-time reduction decreases as the number of records retrieved increases, i.e., the response time remains virtually constant. Again we seem to have evidence to support the hypothesis that, as the size of the response set increases for the same request, the response-time reduction will decrease to a relatively low level (0.1% or less).

Request Number	Response-Time Reduction (A,C)
1	4.49
2	4.03
3	0.92
4	0.32
5	0.47
6	0.12
7	0.50
8	0.23
9	1.07

Table 6. The Response-Time Reduction Between Configurations A and C

5. CONCLUSIONS AND FUTURE WORK

We have shown that the two basic performance claims of the multi-backend database system are valid. While these results are preliminary, they are encouraging. Overall, the response-time improvement ranged from 36.07 percent to 48.94 percent, when the number of backends and their disks is doubled for the same database. The low end of the scale represented a request which involved the actual retrieval of only two records. The high end represents a request which has to access all of the database information. The response-time reductions were also impressive, ranging from a 4.49 percent change to a 0.12 change. In other words, when we double the number of backends and their disks, the response time of a request is nearly invariant despite the fact that the response set for the request is doubled. Another crucial discovery that we made was that the results were consistent and reproducible. The tests were conducted at least twice for most of the request set, with the testing done on different days by different people. The resulting data was consistent and reproducible. *The data presented in this paper represents the last set of* tests for the request set.

The next logical step in the performance evaluation of the multi-backend database system is to extend the testing to include the other request types, update, insert and delete. Additionally, there are still some more tests to run on the retrieval request. We also seek to provide some insight into the internal performance of MBDS. Internal performance measurement provides a microscopic view of the system, by collecting the times of the work distributed and performed by the system components, i.e., in our case, individual processes.

Because MBDS is intended for microprocessor-based backends, winchester-type disks and an Ethernet-like broadcast bus, we will not continue our benchmark work on the present VAX-PDPs configuration. Instead, we plan to download MBDS to either MicroVaxs or Sun Workstations. With either choice, we can utilize a broadcast bus, which was not available when the work began in 1981. We may also eliminate all the physical and virtual memory problems. In the new environment we can perhaps obtain a more thorough benchmarking of MBDS, and study various benchmarking strategies.

ACKNOWLEDGEMENTS

We would like to thank all of those who have helped us with the performance evaluation of the multi-backend database system. Robert C. Tekampe and Robert J. Watson were involved with the development and implementation of the testing software for our system [Teka84]. Prof. Douglas S. Kerr and Dr. Paula R. Strawser provided valuable advice and assistance on the benchmark strategy. Finally, Albert Wong and the technical staff at the Naval Postgraduate School provided assistance with configuring the computer systems for testing.

REFERENCES

[Boyn83a] Boyne, R., et al., "A Message-Oriented Implementation of a Multi-Backend Database System (MBDS)," in *Database Machines*, Leillick and Missikoff (eds), Springer-Verlag, 1983.

[Boyn83b] Boyne, R., et al., "The Implementation of a Multi-Backend Database System (MBDS): Part III - The Message-Oriented Version with Concurrency Control and Secondary-Memory-Based Directory Management," Technical Report, NPS-52-83-003, Naval Postgraduate School, Monterey, California, March 1983.

[Demu84] Demurjian, S. A., et al., "The Implementation of a Multi-Backend Database System (MBDS): Part IV - The Revised Concurrency Control and Directory Management Processes and the Revised Definitions of Inter-Process and Inter-Computer Messages" Technical Report, NPS-52-84-005, Naval Postgraduate School, Monterey, California, March 1984.

[He82] He, X., et al., "The Implementation of a Multi-Backend Database System (MBDS): Part II - The First Prototype MBDS and the Software Engineering Experience," Technical Report, NPS-52-82-008, Naval Postgraduate School, Monterey, California, July 1982; also appeared in *Advanced Database Machine Architecture*, Hsiao (ed), Prentice Hall, 1983.

[Hsia70] Hsiao, D. K., and Harary, F., "A Formal System for Information Retrieval from Files," *Communications of the ACM*, Vol. 13, No. 2, February 1970, Corrigenda, Vol 13., No. 3, March 1970.

[Hsia81a] Hsiao, D.K. and Menon, M.J., "Design and Analysis of a Multi-Backend Database System for Performance Improvement, Functionality Expansion and Capacity Growth (Part I)," Technical Report, OSU-CISRC-TR-81-7, The Ohio State University, Columbus, Ohio, July 1981.

[Hsia81b] Hsiao, D.K. and Menon, M.J., "Design and Analysis of a Multi-Backend Database System for performance Improvement, Functionality Expansion and Capacity Growth (Part II)," Technical Report, OSU-CISRC-TR-81-8, The Ohio State University, Columbus, Ohio, August 1981.

[Kerr82] Kerr, D.S., et al., "The Implementation of a Multi-Backend Database System (MBDS): Part I - Software Engineering Strategies and Efforts Towards a Prototype MBDS," Technical Report, OSU-CISRC-TR-82-1, The Ohio State University, Columbus, Ohio, January 1982; also appeared in *Advanced Database Machine Architecture*, Hsiao (ed), Prentice Hall, 1983.

[Schi84] Schill, J., "Comparative DBMS Performance Test Report," Naval Ocean System Center, San Diego, CA, August 1984.

[Stra84] Strawser, P. R., "A Methodology for Benchmarking Relational Database Machines," Ph. D. Dissertation, The Ohio State University, 1984.

[Teka84] Tekampe, R. C., and Watson, R. J., "Internal and External Performance Measurement Methodologies for Database Systems," Master's Thesis, Naval Postgraduate School, Monterey, California, June 1984.

INITIAL DISTRIBUTION LIST

Defense Technical Information Center Cameron Station Alexandria, VA 22314	2
Dudley Knox Library Code 0142 Naval Postgraduate School Monterey, CA 93943	3
Office of Research Administration Code 012A Naval Postgraduate School Monterey, CA 93943	1
Chairman, Code 52M1 Department of Computer Science Naval Postgraduate School Monterey, CA 93943	20
DAVID K. HSIAO Professor Department of Computer Science Naval Postgraduate School Monterey, CA 93943	150