The Experimental Study of CODASYL Database Administration at the Secondary Storage Level

O. C. AKINYOKUN*
Department of Computer Science, University of Ife, Ile-Ife, Nigeria

Five issues of CODASYL database administration are investigated in this paper. First, the effects of CODASYL storage record placement strategies on the cost of query and update transactions are investigated. Second, the cost of CODASYL database usage pattern is analysed and it is shown that for different usage patterns, different optimal storage structures result. Third, the effects of some changes on an optimal storage structure which the designer may consider desirable are investigated. Fourth, the benefits of clustering storage records which are hierarchically related by CODASYL sets are analysed and shown to decrease with the increase in the levels of the hierarchy. Finally, the effects of statistical variation that may be expected in data populations on the cost of query and update transactions are examined.

Received August 1985, revised August 1986

1. INTRODUCTION

The administration of database at the physical level is concerned with the organisation of data in the secondary storage such that the performance of the database can be efficient. The efficiency of a database can be measured in terms of the cost of its query and update transactions, and secondary storage space. The minimisation of the cost of query transactions may result in additional storage space and increase the cost of update transactions. The selection of a particular storage structure which is favourable to a query or update transaction may cause an increase in the cost of processing some other transactions.

The CODASYL set mode has been discussed in Ref. 39 and the experimental study of its performance presented in Refs 17 and 18. In this research, the performance of CODASYL database is studied by considering storage record placement strategies, the cost of query and update transactions, usage pattern, storage space and statistical variation in data populations.

A placement strategy dictates the structure of a storage record type and provides the mechanism for accessing the occurrences of the record type.[11] The placement strategies are calc, sequential, clustered via set and clustered via set near owner.

The calc placement strategy permits the occurrences of a record type to be stored in the addresses generated randomly using the record key. It provides the efficient retrieval and update of individual record occurrences using the record key. The sequential placement strategy permits the occurrences of a record type to be stored in a specified range of pages in a sequence of ascending or descending values of a defined key. It optimises the processing of the record occurrences in the specified key sequence. The clustered via-set placement strategy permits the member records of a set occurrence to be stored together in a sequence within a specified page range. The clustered via set near owner permits the member records of a set occurrence to be stored within the page containing the owner record. In a situation where the set occurrence is too large to be stored within a page, the member records which cannot be stored within this page are stored on a page adjacent to that page which contains the owner record. In general, the choice of placement strategy usually optimises one type of access at the expense of others.

The first phase of our study addresses the optimisation of the storage structure of CODASYL database. One intention is to demonstrate the effects of CODASYL database storage record placement strategies on the cost of query and update transactions. The other intention is to show that for different database usage patterns different optimal storage structures result.

The second and final phase is concerned with the sensitivity analysis of the optimal storage structure obtained in phase one. Three issues are addressed in this phase. The effects of some changes on an existing optimal storage structure which are favoured by the designer are investigated. The CODASYL storage record placement strategy described as 'clustered via set near owner' is intended to minimise the cost of database processing, especially when the owner and member records of a set occurrence are always required together. It is shown in this study that in a hierarchy of sets, the benefits of clustering storage records decrease with the increase in levels of the hierarchy. Finally, the effects of the statistical variation that may be expected in data populations on the cost of query and update transactions are investigated.

The organisation of the paper is as follows. The results of the investigations carried out in the first phase of the experiment are presented in Section 2. Section 3 presents the results of the investigations carried out in the second phase. Some conclusions are drawn in Section 4.

2. THE OPTIMISATION OF CODASYL DATABASE STORAGE STRUCTURE

The procedure for the optimisation of CODASYL database storage structure has five modules. These are: the cluster matrix generator, the CODASYL storage area definer, the query and update *program* generator, the query and update cost evaluator and the optimal

* Author now at: Department of Mathematics and Computer Technology, Federal University of Technology, Akure, Nigeria.

storage structure selector. In this section, the assumptions made in the experiment are outlined and the primary inputs described briefly. Furthermore, the modules are described in some detail and the results obtained from the case study of a conference database presented.

2.1 The basic assumptions

The following assumptions are made in the experiment.

- The set mode described as embedded pointer array is assumed.
- A schema record type maps to only one storage record type.
- A fixed-size page environment in which the unit of transfer of data from the secondary storage to the memory is a page is assumed.
- The set order is assumed sorted and indexes supported by the SCOPE Indexed Sequential (SIS) file mechanism as described in Ref. 48.
- The probability of overflow of calc chain to another page is neglected.

2.2 The primary inputs

The primary inputs to the experiment are the CODASYL schema, the statistics, query and update transactions, and usage pattern of the intended database.

A CODASYL schema is generally described as a graph where the vertices and the edges of the graph represent record types and set types respectively. For convenience in our experimental study, a schema is represented by a square matrix where the rows and the columns are defined by record types. An entry of the schema matrix is represented by $e_{i,j}$, where $e_{i,j}$ is defined by:

$$e_{i,j} = \begin{cases} 0 & \text{if record type } i \text{ equals record type } j \\ 1 & \text{if record type } i \text{ is a member of a set owned by record type } j \\ \not\emptyset & \text{otherwise} \end{cases}$$

The relational representation of the intended database statistics, query and update transactions, and usage pattern is adopted. A typical representation for the case study of conference database is presented in Appendices A.2 and A.3.

2.3 The cluster matrix generation

One of the ways by which the efficient performance of the CODASYL database can be improved is by clustering storage records, which will often be required together by the database users.[34] Mathematical algorithms for clustering data items and record types have been proposed in Refs 23 and 41. It is very unlikely that formal deterministic procedures for clustering data will yield a satisfactory answer to the problem of the implementation of a large database.[46] In this study, therefore, a heuristic approach for the generation of the feasible clusters of CODASYL storage record types is chosen.

A feasible cluster of CODASYL storage record types is represented by a matrix. An entry of the matrix is denoted by $C_{i,j}$, which describes the placement strategy of a member record type i of a set type owned by a record type j. In general, $C_{i,j}$ is defined by:

$$C_{i,j} = \begin{cases} 0 & \text{if } i \text{ is calc'ed or } i \text{ is randomly placed relative to } j \\ 1 & \text{if } i \text{ is sequential} \\ 3 & \text{if } i \text{ is clustered via a set owned by } j \\ 4 & \text{if } i \text{ is clustered via a set near the owner } j \\ \not\emptyset & \text{otherwise} \end{cases}$$

2.3.1 The rules for clustering storage structures

In this work, the record clustering rules proposed in Ref. 27 are modified by considering access methods and database usage pattern. One of the rules follows from the logical restrictions of CODASYL database design. The others are introduced as desirable mechanisms to reduce the size of the solution space to be examined.

2.3.1.1 The clustering rule one

Suppose a record type R_1 is a member of two set types S_1 and S_2, say, where the owner record types of S_1 and S_2 are R_2 and R_3, respectively. Then R_1 may be clustered via S_1 or S_2 but not both. Alternatively R_1 can be clustered via S_1 near R_2 or via S_2 near R_3, but not both. For an effective and efficient design, the expected number of members of an occurrence of either of the two set types together with their frequencies of traversal are desirable for the determination of the set type to select. This rule can be generalised to the situation when R_1 is a member of three or more set types.

2.3.1.2 The clustering rule two

Let a record type R_1 be a member of the set type S_1 owned by the record type R_2. Suppose R_1 is the owner of a set type S_2 whose member record type is R_3. If R_1 is clustered via S_1 near R_2, then R_3 cannot be clustered via S_2 near R_1. R_3 can be clustered via S_2 near R_1 only if the result of a trade-off analysis favours it.

2.3.1.3 The clustering rule three

Let a record type R_1 be the owner of set types $S_2, S_3, \ldots, S_n$ and the corresponding member record types are $R_2, R_3, \ldots, R_n$. Then at least one of these member record types may be clustered via its corresponding set type near R_1.

2.3.1.4 The clustering rule four

Let a record type R_1 be a member of the set type S_1 owned by the record type R_2. Suppose R_1 is the owner of a set type S_2 whose member record type is R_3. If R_1 is clustered via S_1, R_3 may be clustered via S_2 as well.

2.3.1.5 The clustering rule five

Let a serial query represented by T_1 with frequency m be defined on a record type R_1. A serial query is that whose accesses of the occurrences of a record type are sequential. Suppose there are some query and/or update transactions represented by T_2, T_3, T_4, T_5 and T_6, say, each of which involves R_1 and some other record types R_2, R_3 and R_4.

Let the total frequency of these transactions be n ($m \gg n$). Suppose R_1 is the owner of the set types S_2 and S_3, whose members are R_2 and R_3, respectively. if the cost of T_1 is P page accesses and the sum of the cost of T_2, T_3, T_4, T_5 and T_6 is Q page accesses ($P \gg Q$), then R_2 and R_3 may not be clustered via S_2 and S_3 respectively near R_1.

The inputs to the cluster matrix generator are the schema matrix, database statistics, usage pattern, query and update transactions.

2.3.2 The cluster matrices generated for conference database

Applying the rules for clustering storage records on the conference database described in Appendix A, sixteen cluster matrices are generated. In order to minimise the size of this paper, we report only the cluster matrix shown in Table 2.1. However, due reference will be made to some other matrices when the need arises in the remaining part of this paper.

Table 2.1. The cluster matrix of a storage structure

	Owner										
Member	1	2	3	4	5	6	7	8	9	10	11
1	0	—	—	—	—	—	—	—	—	—	—
2	4	0	—	—	—	—	—	—	—	—	—
3	4	—	0	—	—	—	—	—	—	—	—
4	—	—	—	0	—	—	—	—	—	—	—
5	—	3	—	—	—	—	—	—	—	—	—
6	—	—	3	—	—	—	—	—	—	—	—
7	—	—	—	4	—	—	0	—	—	—	—
8	—	—	—	4	0	—	—	0	—	—	—
9	—	—	—	—	—	—	3	0	—	—	—
10	—	3	—	—	—	—	—	0	—	—	—
11	—	—	3	—	—	—	—	0	—	—	—

In Table 2.1 the $C_{1,1}$ indicates that the placement of the record type *person* is calc. The entry $C_{2,1}$ indicates that the record type *author* is clustered via the set type *persauth* near *person*. The other entries can be described in a similar manner using the definition of $C_{i,j}$ given above.

2.4 The CODASYL area definition

In this experiment, the formula for calculating the total number of pages denoted by N per CODASYL area proposed in Ref. 25 is adopted. The formula is:

$$N = \frac{100n(L+2)}{D(P-8)}$$

where P represents the page size in bytes, D represents the percentage packing density, n represents the total number of records to be stored, L represents the average record length in bytes. Each record receives a 2-word line index and each page has a 4-word header and a 4-word footer used by the system.

The inputs to the area definer are the database statistics and the cluster matrices. An area is described by the number of pages that composed it and the storage record types whose occurrences are contained within it.

2.4.1 The CODASYL area generated for conference database

In the experimental study, the record size is assumed fixed and the page size is 1,024 bytes. The percentage packing density is 80. The definition of the areas obtained for the storage structure depicted in Table 2.1 is described in Tables 2.2 and 2.3.

Table 2.2. The record types per area

CODASYL area name	Storage record type
person-author-referee-area	person
person-author-referee-area	author
person-author-referee-area	referee
conference-session-paper-area	conference
conference-session-paper-area	session
conference-session-paper-area	paper
presenter-area	presenter
phone-area	phone
accpaper-area	accpaper
authorship-area	authorship
refereeing-area	refereeing

Table 2.3. The pages per area

CODASYL area name	No. of pages
person-author-referee-area	1500
conference-session-paper-area	520
presenter-area	400
phone-area	200
accpaper-area	100
authorship-area	2000
refereeing-area	2000

2.5 The query and update program generation

The techniques for the evaluation of query program on CODASYL database are proposed in Ref. 14. That technique is modified to cover update transactions and implemented in this experiment using a variant of the algorithm which determines all the distinct spanning trees of a graph. The inputs to the query and update program generator are the CODASYL schema matrix, database statistics, query and update transactions. A query or update program is described by a record type which acts as the entry point or root to the database and the remaining record types involved in the transaction in the sequence of visiting them.

2.5.1 The programs generated for conference database

In order to minimise the size of this paper we present only some of the programs generated for query Q_6 and update transaction U_1.

Query Q_6 has only one entry point but two feasible programs. The descriptions of these programs are:

{[$Q_{6,1,1}$]; conference, session, paper}
{[$Q_{6,1,2}$]; conference, paper, session}

The first program prescribes the navigation of the database as follows. Access the given occurrence xyz, say of the record type *conference*. Following this, access the occurrences of the record type *session* which are owned by xyz in the set type *confsess*. Finally, access the occurrences of the record type *paper* which are owned by xyz in the set type *confpape*. The second programme prescribes the navigation of the occurrence xyz of *conference*, then the occurrences of *paper* before the occurrences of *session*.

Two of the programs generated for U_1 are:

{[$U_{1,1,1}$]; paper, accpaper, authorship, refereeing, presenter, conference}

{[$U_{1,1,2}$]; paper, conference, presenter, accpaper, authorship, refereeing}

The first program prescribes the navigation of the database as follows. Access and delete the given occurrence xyz, say, of the record type *paper*. Delete the occurrences of the record type *accpaper* owned by xyz in the set type *papeaccp*. Delete the occurrences of the record types *authorship* and *refereeing* owned by xyz. Disconnect the record xyz from its owners in the set types *confpape* and *prespape*.

2.6 The query and update cost evaluation

Query access cost model for CODASYL database is proposed in Ref. 14. The query and update cost model proposed in Ref. 26 does not consider the CODASYL storage record placement strategy described as clustered via set near owner. In this experiment, the two cost models referenced above are combined and modified to cater for the placement strategy described as clustered via set near owner.

The inputs to the cost evaluator are: the cluster matrices, query and update programs, and the CODASYL area. The cost of a transaction is evaluated in four steps. First, the cost of accessing an area involved in the transaction is calculated. Second, this cost is multiplied by the number of times the area is visited. Third, the first and second steps are repeated for the other areas involved in the transaction. Fourth, the overall costs of the areas are added together, and multiplied by the frequency of the transaction.

2.6.1 The cost of the conference database

The cost of the transactions presented in Appendix A.3 for the sixteen storage structures generated for the conference database is shown in Table 2.4. The major factor which is responsible for the differences in the cost from one storage structure to another is the variation in the placement strategies of the record types which are involved in each transaction.

2.7 The optimal storage structure selection

The inputs to the optimal storage structure selector are the query and update cost matrix, cluster matrix and the CODASYL area. The selection of the optimal storage structure is in three steps. First, the minimum cost of each transaction per storage structure is obtained. It is noted that the cost of the programs of a transaction may vary from one to another. Second, the sum of the minimum cost for all transactions per storage structure is calculated. Finally, the storage structure with the least cost is chosen as the optimal.

2.7.1 The optimal storage structure for conference database

Seven usage patterns of the conference database are formulated by varying the frequencies of the transactions defined in Appendix A.3. In this paper, only the results obtained for two of the usage patterns will be presented and analysed.

2.7.1.1 Usage pattern one

In usage pattern one, the frequency of each of the transactions defined in Appendix A.3 is assumed to be one. The cost of this usage pattern for the sixteen storage structures is shown in Table 2.5.

Table 2.4. The cost of database transactions

Storage structure	Query transactions										Update transactions			
	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_7	Q_8	Q_9	Q_{10}	U_1	U_2	U_3	U_4
1	6	3	12	53	53	53	13	28	853	6	91	64	2	6
2	6	3	42	53	53	53	13	28	253	3	88	64	2	6
3	3	6	12	53	53	53	13	13	853	6	88	64	2	6
4	3	6	42	53	53	53	13	13	253	3	85	64	2	6
5	6	3	12	53	53	53	17	32	853	6	91	64	2	4
6	6	3	42	53	53	53	17	32	253	3	88	64	2	4
7	3	6	12	53	53	53	17	17	853	6	88	64	2	4
8	3	6	42	53	53	53	17	17	253	3	85	64	2	4
9	6	3	12	4	202	204	13	28	1003	6	91	14	2	6
10	6	3	42	4	202	204	13	28	403	3	88	14	2	6
11	3	6	12	4	202	204	13	13	1003	6	88	14	2	6
12	3	6	42	4	202	204	13	13	403	3	85	14	2	6
13	6	3	12	4	202	204	17	32	1003	6	91	14	2	4
14	6	3	42	4	202	204	17	32	413	3	88	14	2	4
15	3	6	12	4	202	204	17	17	1003	6	88	14	2	4
16	3	6	42	4	202	204	17	17	403	3	85	14	2	4

In Table 2.5 the storage structure represented by 4 has the least cost; hence it is the optimal. A simplified form of the description of this structure using the Database Storage Description Language (DSDL) proposed in Ref. 11 is as follows:

```
STORAGE SCHEMA conference-database,
 REPRESENT ALL,
   STORAGE AREA person-author-referee-
     area,
    INITIAL SIZE 1500 PAGES,
    FRAME SIZE 1024 CHARACTERS;
   STORAGE AREA conference-session-
     paper-area,
    INITIAL SIZE 520 PAGES,
    FRAME SIZE 1024 CHARACTERS;
   STORAGE AREA presenter-area,
    INITIAL SIZE 400 PAGES
    FRAME SIZE 1024 CHARACTERS;
   STORAGE AREA phone-area,
    INITIAL SIZE 200 PAGES,
    FRAME SIZE 1024 CHARACTERS;.
   STORAGE AREA accpaper-area,
    INITIAL SIZE 100 PAGES,
    FRAME SIZE 1024 CHARACTERS;
   STORAGE AREA authorship-area,
    INITIAL SIZE 2000 PAGES,
    FRAME SIZE 1024 CHARACTERS;
   STORAGE AREA refereeing-area,
    INITIAL SIZE 2000 PAGES,
    FRAME SIZE 1024 CHARACTERS;
   STORAGE RECORD Person,
    PLACEMENT calc USING pnumber;
   STORAGE RECORD author,
    PLACEMENT clustered VIA persauth
      NEAR person;
   STORAGE RECORD referee,
    PLACEMENT clustered VIA persrefe
      NEAR person;
   STORAGE RECORD conference,
    PLACEMENT calc USING confname;
   STORAGE RECORD presenter,
    PLACEMENT clustered VIA authpres;
   STORAGE RECORD phone,
    PLACEMENT clustered VIA refephon;
   STORAGE RECORD session,
    PLACEMENT clustered VIA confsess
      NEAR conference;
   STORAGE RECORD paper,
    PLACEMENT clustered VIA confpape
      NEAR conference;
   STORAGE RECORD accpaper,
    PLACEMENT clustered VIA SESSACCP;
   STORAGE RECORD authorship
    PLACEMENT clustered VIA papeauth;
   STORAGE RECORD refereeing,
    PLACEMENT clustered VIA paperefe,
END STORAGE SCHEMA
```

2.7.1.2 Usage pattern two

In usage pattern two, the frequencies of queries Q_1, Q_2, Q_8 and Q_9 are 10, 100, 10 and 10, respectively. The frequencies of the other transactions are taken to be one.

Table 2.5. The access cost of usage pattern one

Storage structure	Cost of transactions
1	1243
2	667
3	1225
4	649
5	1249
6	673
7	1231
8	655
9	1594
10	1018
11	1576
12	1000
13	1600
14	1024
15	1572
16	1006

Table 2.6. The access cost of usage pattern two

Storage structure	Cost of transactions
1	9523
2	3547
3	9640
4	3664
5	9565
6	3589
7	3676
8	3706
9	11224
10	5248
11	10341
12	5365
13	11268
14	5290
15	10383
16	5407

The cost of this usage pattern for the sixteen storage structures is shown in Table 2.6.

In Table 2.6, the storage structure represented by 2 has the least cost, hence it is the optimal storage structure. In this structure, the record type *authorship* is clustered via set type *authauth*. It is noted that *authorship* is clustered via *papeauth* in the optimal structure obtained for usage pattern one. It is concluded therefore that if the frequency of query Q_2 is large compared with that of query Q_1, there is a benefit when *authorship* is clustered via *authauth*.

3. THE SENSITIVITY ANALYSIS OF OPTIMAL STORAGE STRUCTURE

Three issues are addressed in this section. First, the analysis of the cost of a storage structure which is obtained due to certain changes in the placement strategies of the record types of the existing optimal structure is carried out. Second, the cost and benefits of clustering CODASYL storage records are investigated. Finally, the cost of the statistical variation that may be expected in data populations is investigated.

3.1 The cost effects of changes in record placement strategies

The existing optimal storage structure is analysed and the placement of some record types is changed. The new storage structure is evaluated with the objective of determining whether a new optimal structure will result. The optimal storage structure obtained for usage pattern two, described in the previous section, is considered for this investigation and the following changes are made:

(*a*) the record type *paper* is calc'ed;
(*b*) the record type *authorship* is clustered via *papeauth* near *paper*;
(*c*) the record type *session* is clustered via *confsess*.

It is remarked that a new CODASYL area definition is obtained because of these changes. The cost of usage pattern two on the new storage structure is shown in Table 3.1. It is observed that while the cost of transactions Q_1, Q_4, Q_{10}, U_1 and U_2 is reduced, the cost of the other transactions is increased. The overall cost of the new storage structure is 5,320 page accesses. This shows an increase of 1,773 page accesses when compared with that of the existing optimal storage structure. Considering the existing transactions, therefore, it is not a wise design decision to calc the record type paper.

3.2 The cost effects of multiple clustering of storage records

Clustering rule two, described in Section 2.3, allows for at most two-level clustering of storage records. In this section the rule is relaxed such that multiple-level clustering of storage records is allowed. The cost of the multiple-level clustering for some CODASYL set trees in the conference database is analysed in the following.

Consider the optimal storage structure obtained for usage pattern two. Suppose the record type *refereeing* is clustered via *paperefe* near *paper*, and *paper*, in turn, is clustered via *confpape* near *conference*. A new CODASYL area definition is obtained as a result of this decision. The cost of the new storage structure is given in Table 3.2 and the overall cost is 4,094 page accesses. This shows an increase of 547 page accesses when compared with the cost of the existing optimal storage structure.

3.3 The cost effects of statistical variation of data population

The effects of the variation in the expected number of members of an occurrence of a set type on set mode have been discussed in Ref. 39 and the experimental study of these carried out and reported in Refs 17 and 18. In our study, an existing optimal storage structure is analysed and the expected number of the member records of the occurrences of some set types is varied. The cost of the database transactions on the new storage structure which results is analysed.

Consider the optimal storage structure obtained for usage pattern two. With reference to the database statistics of Appendix A.2.3, the following changes are made:

(*a*) the expected number of occurrences of *refereeing* per occurrence of the set type *paperefe* is 2;
(*b*) the expected number of occurrences of *refereeing* per occurrence of the set type *referefe* is 80.

The cost of usage pattern two on the new storage structure is given in Table 3.3. The overall cost is 3,572 page accesses and shows an increase of 25 page accesses when compared with the cost of the existing optimal storage structure. In order to reduce this cost, the following is desirable:

(*a*) change the placement strategy of *refereeing* from clustered via *paper* to clustered via *referefe*.

This change will force a reduction in the overall cost obtained above. In particular, the cost of transaction Q_3 will be reduced by 60 page accesses.

4. THE CONCLUSION

Some basic assumptions have been made in the framework for the CODASYL database administration presented in this paper. The assumptions can be dropped in order to carry out further experimental study.

Table 3.1. The cost of the new storage structure

Transactions	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_7	Q_8	Q_9	Q_{10}	U_1	U_2	U_3	U_4
Cost	10	600	42	2	202	203	13	80	4010	2	84	64	2	6

Table 3.2. The cost of the relaxed storage structure

Transactions	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_7	Q_8	Q_9	Q_{10}	U_1	U_2	U_3	U_4
Cost	60	300	42	4	249	251	13	280	2490	249	84	64	2	6

Table 3.3. The cost of variation in data population

Transactions	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_7	Q_8	Q_9	Q_{10}	U_1	U_2	U_3	U_4
Cost	60	300	82	53	53	53	13	289	2530	3	73	64	2	6

The complex structures such as multimember set type have not been considered, particularly in the case study of the conference database. A multimember set type can be handled in the experiment by breaking it into two or more set types. This mechanism has been proposed in Ref. 39 for the implementation of multimember set type. The set mode such as chain next and prior; chain next, prior and owner can be considered. Furthermore, allowance can be given for a schema record type to map to two or more storage record types.

In general, the procedure for our experiment is modular, and may be used for the interactive design of the storage structure of CODASYL database. The procedure might more usefully be applied to the reorganisation of an existing CODASYL database, where the designer could first contrast the theoretical results with those actually obtained.

Acknowledgement

This research was carried out in the University of East Anglia while on a Ph.D programme sponsored by the Commonwealth Scholarship Commission, United Kingdom. The technical guidance and support of my supervisor, Professor P. M. Stocker, are gratefully acknowledged. My thanks to Dr P. A. Dearnley of the University of East Anglia and Dr P. M. D. Gray of the University of Aberdeen for their contributions. My sincere appreciation of the comments of the referees as well.

REFERENCES

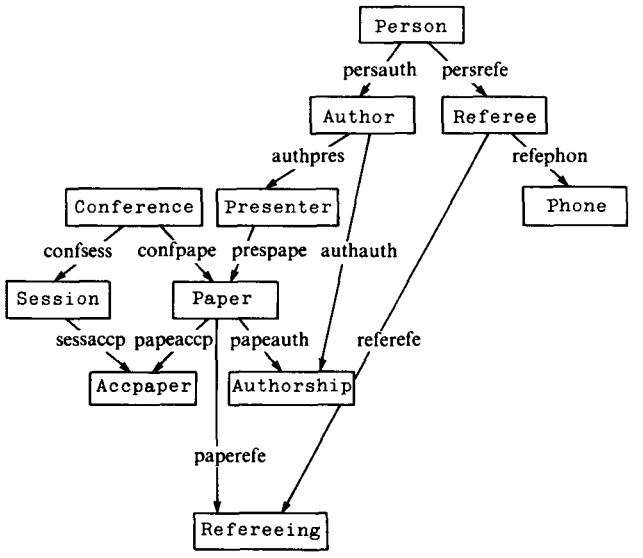
1. O. C. Akinyokun, A methodology for the automatic design of database systems, *Ph.D. Thesis*, University of East Anglia, England (1984).
2. E. Berelian, A methodology for database design in a paging environment, *Ph.D. Thesis*, University.of Michigan, USA (1977).
3. P. A. Bernstein *et al.*, Query processing in a system of distributed databases (SDD-1). *ACM Transactions on Database Systems*, **6** (4), 602–624 (1981).
4. H. M. Blanken, Automatic generation of storage structures for a DBTG data base system. *Proceedings of the 1st International Conference on Databases*, pp. 99–118 (1980).
5. J. A. Bubenko *et al.*, From information requirements to DBTG-Data structures. *Proceedings of ACM SIGMOD/SIGPLAN International Conference*, pp. 73–85 (1976).
6. J. V. Carlis *et al.*, Physical database design; a DSS approach, *Information and Management*, **6**, 211–244 (1983).
7. E. K. Clemons, Rational data base standard: an examination of the 1978 CODASYL DDLC report. *Information Systems*, **4**, 235–239 (1979).
8. CODASYL DBTG, The report of the CODASYL Data Base Task Group (1971).
9. CODASYL DDLC, The report of the CODASYL Data Description Language Committee (1973).
10. CODASYL DDLC, The report of the CODASYL Data Description Language Committee (1978).
11. CODASYL DDLC, The report of the CODASYL Data Description Language Committee (1981).
12. Cullaine IDMS, Integrated Database Management System (IDMS) Publications. *Data Definition Languages, Utilities and GCI Reference Guide*, release 3.1; *Data Manipulation Language Programmer's Reference Guide*, release 3.1, Cullaine Corporation, Boston, USA (1975).
13. R. Dahl and J. A. Bubenko, IDBD: an interactive design tool for CODASYL-DBTG-type databases. *Proceedings of the International Conference on Very Large Data Bases*, pp. 108–121 (1982).
14. U. Dayal and N. Goodman, Query optimization for CODASYL database systems. *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 138–150 (1982).
15. P. A. Dearnley, Physical design of databases. In *Databases: Role and Structure*, edited P. M. Stocker, P. M. D. Gray and M. P. Atkinson pp. 81–91. Cambridge University Press (1984).
16. B. C. M. Douque and G. M. Nijssen (eds.), *Proceedings of the IFIP TC-2 Special Working Conference on Data Base Description*. North-Holland, Amsterdam (1975).
17. W. Effersberg *et al.*, An experiment in learning DBTG database administration. *Information Systems* **5**, 137–147 (1980).
18. W. Effersberg *et al.*, Measurement and evaluation of techniques for implementing COSETS – a case study. *Proceedings of the International Conference on Databases*, pp. 135–159 (1980).
19. T. J. Gambino and R. Gerritsen, A data base design decision support system. *Proceedings of the International Conference on Very Large Data Bases*, pp. 534–544 (1977).
20. R. Gerritsen, A preliminary system for the design of DBTG data structures. *Communications of the ACM* **18** (10), 551–557 (1975).
21. P. M. D. Gray, Use of automatic programming and simulation to facilitate operations on CODASYL databases. *INFOTECH State of the Art Report on Databases*, series 9, no. 8, pp. 347–469 (1981).
22. P. M. D. Gray, The functional data model related to the CODASYL model. In *Databases – Role and Structure*, edited P. M. Stocker, P. M. D. Gray and M. P. Atkinson, pp. 57–79. Cambridge University Press (1984).
23. J. A. Hoffer and D. G. Severance, The use of cluster analysis in physical database design. *Proceedings of the International Conference on Very Large Data Bases*, pp. 69–86 (1975).
24. Honeywell IDS-II, Integrated Data Store II (IDS-II) Publications: *DM-IV Database Administrator Reference Manual, DM-IV COBOL-74 Programmer Reference Manual.* Honeywell Information Systems Inc.
25. ICL Dataskil IDMX, Integrated Database Management System (IDMS) Publications: *IDMS Data Administration Languages Reference Manual, IDMS COBOL DML Reference Manual, IDMS Utilities Reference Manual, IDMS Sizing Reference Manual.* International Computers Limited Dataskil. Reading, U.K. (1978).
26. H. K. Jain, A comprehensive model for the storage structure design of CODASYL database. *Information Systems* **9** (34), 217–230 (1984).
27. R. H. Katz and E. Wong, Resolving conflicts in global storage design through replication. *ACM Transactions on Database Systems* **6** (1), 33–42 (1983).
28. J. S. Knowles and D. M. R. Bell, The CODASYL model. In *Databases – Role and Structure*, edited P. M. Stocker, P. M. D. Gray and M. P. Atkinson, pp. 19–56. Cambridge University Press (1984).
29. Logica Rapport-3, Rapport Relations DBMS Publications: *Rapport-3 Designing and Using a Database Reference Manual.* Logica Ltd, U.K. (1982).
30. M. E. S. Loomis, The '78 CODASYL database model: a comparison with preceding specifications. *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 30–44 (1980).
31. M. E. S. Loomis and F. W. Allen, Paging behaviour and performance optimization in a CODASYL DBMS. *Pro-*

ceedings of the International Conference on Databases, pp. 119–135 (1980).
32. F. A. Manola, A review of the 1978 CODASYL database specifications. *Proceedings of the International Conference on Very Large Data Bases*, pp. 232–242 (1978).
33. F. A. Manola, An evaluation of the new CODASYL and ANSI/SPARC database proposals. *INFOTECH State-of-the-Art Report on Database Technology* **2**, 133–151 (1978).
34. S. T. March, Techniques for structuring database records. *Computing Surveys* **15** (1), 45–79 (1983).
35. A. S. Michaels *et al.*, A comparison of the relational and CODASYL approaches to database management. *Computing Surveys* **8** (1), 125–151 (1976).
36. M. F. Mitoma, Optimal data base schema design. *Ph.D. Thesis*, University of Michigan, U.S.A. (1975).
37. G. M. Nijssen, Two major flaws in the CODASYL DDL 1973 and proposed corrections. *Information Systems* **1**, 115–132 (1975).
38. M. L. O'Connel, The 1978 CODASYL database specifications. *INFOTECH State-of-the-Art Report on Database Technology* **2**, 259–168 (1978).
39. T. W. Olle, *The CODASYL Approach to Database Management*. John Wiley, Chichester (1980).
40. T. W. Olle *et al.* (eds), Information System Design Methodologies; A Comparative Review. North-Holland, Amsterdam (1982).
41. M. Scholnick, A clustering algorithm for hierarchical structure. *ACM Transactions on Database Systems* **2** (1) 27–44 (1977).
42. P. G. Selinger *et al.*, Access path selection in a relational database management system. *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 23–34 (1979).
43. J. M. Smith *et al.*, MULTIBASE: Integrating heterogeneous distributed database systems. *Proceedings of the National Computer Conference, U.S.A.*, pp. 487–499 (1981).
44. W. Staniszkis *et al.*, Physical data base design for CODASYL DBMS. *Methodology and Tools for Data Base Design* pp. 119–148. North-Holland, Amsterdam (1983).
45. W. Staniszkis *et al.*, Probabilistic approach to evaluation of data manipulation algorithms in CODASYL data base environment. *Proceedings of the International Conference on Databases*, pp. 332–357 (1983).
46. P. M. Stocker, The structuring of databases at the implementation level. In *Architecture and Models in Data Base Management Systems*, pp. 261–276. North-Holland, Amsterdam (1977).
47. T. J. Teorey and J. P. Fry, *Design of Database Structures*. Prentice-Hall, Englewood Cliffs, NJ, U.S.A. (1982).
48. J. P. Tremblay and P. G. Sorenson, *An Introduction to Data Structures with Applications*. McGraw-Hill International Book Co., 2nd ed. (1985).
49. K. Y. Whang *et al.*, Physical design of network model databases using the property of separability. *Proceedings of the International Conference on Very Large Data Bases*, pp. 98–107 (1982).

APPENDIX A

The CODASYL conference database statistics and usage

A.1 The CODASYL schema graph



A.2 The statistics of CODASYL conference database

A.2.1 The statistics of record types

Record no.	Name	Population	Size
1	PERSON	15000	200
2	AUTHOR	2000	100
3	REFEREE	200	100
4	CONFERENCE	10	150
5	PRESENTER	400	100
6	PHONE	400	100
7	SESSION	100	150
8	PAPER	2000	200
9	ACCPAPER	500	100
10	AUTHORSHIP	8000	200
11	REFEREEING	8000	200

A.2.2 The statistics of data items

Record no.	D-item no.	D-item name	Value type	D-item sel
4	1	CONFNAME	10	0.10000
4	2	CONFPLACE	10	0.10000
4	3	STARTDATE	10	0.10000
4	4	ENDDATE	10	0.10000
7	5	STITLE	100	0.01000
7	6	SDATE	20	0.05000
7	7	SPLACE	10	0.10000
7	8	STARTTIME	3	0.33333
7	9	ENDTIME	3	0.33333
8	10	PATITLE	2000	0.00050
8	11	DATEACKSENT	100	0.01000
8	12	REGDATE	100	0.01000
8	13	DECDATE	100	0.01000
5	14	PNUMBER	400	0.00250
2	15	PNUMBER	2000	0.00050
3	16	PNUMBER	200	0.00500
11	17	REFREPORT	8000	0.00013
11	18	REFEREPNO	8000	0.00013
1	19	PNUMBER	15000.	0.00007
1	20	PNAME	7500	0.00013
1	21	PADDRESS	15000	0.00007
6	22	PHONENUM	200	0.00500

A.2.3 The statistics of set types

Set-no.	Set-name	Own-no.	Mem-no.	Mem/own
1	PERSAUTH	1	2	1
2	PERSREFE	1	3	1
3	AUTHPRES	2	5	1
4	REFEPHON	3	6	2
5	CONFSESS	4	7	10
6	CONFPAPE	4	8	200
7	PRESPAPE	5	8	5
8	AUTHAUTH	2	10	4
9	REFEREFE	3	11	40
10	SESSACCP	7	9	5
11	PAPEACCP	8	9	1
12	PAPEREFE	8	11	4
13	PAPEAUTH	8	10	4

A.3 The transactions on CODASYL conference database

The transactions presented in the following do not form an exhaustive list of the users' activities on the conference database. However, it is adequate for the investigations carried out in the study.

Q_1 Find the authors of a given paper.
Q_2 Find the papers written by a given author
Q_3 Find the refereeing report of a given referee
Q_4 Find the sessions of a given conference
Q_5 Find the papers submitted to a given conference
Q_6 Find the sessions and the papers of a given conference
Q_7 Find the presenters of the papers to be presented in a given session
Q_8 Find the authors of the papers to be presented in a given session
Q_9 Find the reports of the referees for the papers submitted to a given conference
Q_{10} Find the reports of the referee for a given paper
U_1 Delete a given paper
U_2 Store a given paper
U_3 Modify the address of a given person
U_4 Connect a given accpaper to a session

A.3.1 The type-one transaction definition

Transaction no.	Number of record type	Entry point of transaction	Frequency of transaction
Q_1	2	PAPER	1
Q_2	2	AUTHOR	1
Q_3	2	REFEREE	1
Q_4	2	CONFERENCE	1
Q_5	2	CONFERENCE	1
Q_6	3	CONFERENCE	1
Q_7	4	SESSION	1
Q_8	4	SESSION	1
Q_9	3	CONFERENCE	1
Q_{10}	2	PAPER	1
U_1	6	PAPER	1
U_2	3	PAPER	1
U_3	1	PERSON	1
U_4	2	SESSION	1

A.3.2 The type-two transaction definition

The table below gives the description of the transactions Q_{1-7}. The other transactions can be described in a similar way.

Transaction no.	Record name	Access mode	Selectivity
Q_1	PAPER	calc	0.00050
Q_1	AUTHORSHIP	set	0.00050
Q_2	AUTHOR	calc	0.00050
Q_2	AUTHORSHIP	set	0.00050
Q_3	REFEREE	calc	0.00500
Q_3	REFEREEING	set	0.00500
Q_4	CONFERENCE	calc	0.10000
Q_4	SESSION	set	0.10000
Q_5	CONFERENCE	calc	0.10000
Q_5	PAPER	set	0.10000
Q_6	CONFERENCE	calc	0.10000
Q_6	SESSION	set	0.10000
Q_6	PAPER	set	0.10000
Q_7	SESSION	calc	0.01000
Q_7	ACCPAPER	set	0.01000
Q_7	PAPER	set	0.00250
Q_7	PRESENTER	set	0.01250