数理解析研究所講究録
第353巻 1979年 20-29

20

Query Processing in a Relational Database Using Abstracted Characteristics of Data

Le Viet Chung*, Yahiko Kambayashi*,
Katsumi Tanaka** and Shuzo Yajima*

* Department of Information Science, Kyoto University
** College of Liberal Arts, Kobe University

1. Introduction

In the database field, the following problems are important:

(1) to provide efficient processing of retrieval/update operations, and

(2) to provide a high level user interface.

In this paper, in order to provide a solution for these problems, we present a new concept called "abstracted characteristics of data". Fig.1(a), 1(b) and 1(c) describe the difference between conventional databases and relational databases which use abstracted characteristics.

Fig.1(a) shows a simplified diagram of conventional relational databases. In the figure, security constraints specify time-independent conditions(characteristics) that must be satisfied by base relations in order to protect the data against invalid update operations. Functional and multivalued dependencies are important static constraints. Triggers(or demons) are defined as dynamic constraints. Relations consist of extensional and intensional data. The former are sets of tuples actually stored as base relations. The latter are derived by extensional data and axioms[MINK7803]. In this paper, we deal with extensional data and some class of axioms called views[CHAMG7505].

Fig.1(b) shows partly the relational databases with abstracted characteristics stored. Abstracted characteristics are defined as information abstracted from sets of tuples. They are classified into:

[A] Time-independent characteristics vs time-dependent characteristics, and

[B] Global characteristics vs local characteristics.

In this paper, we discuss only characteristics which are easily computed and managed. Among them are functional dependencies, time-dependent functional dependencies, bound numbers of associated values, attribute ranges and other inner-/inter-relation constraints. A classification of abstracted characteristics can be found in [LE-K7902].

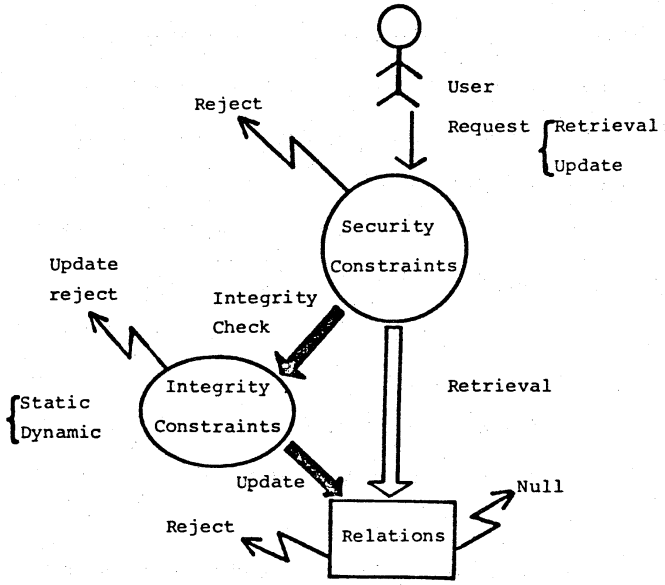


Fig.1(a) Simplified Diagram of Relational Databases

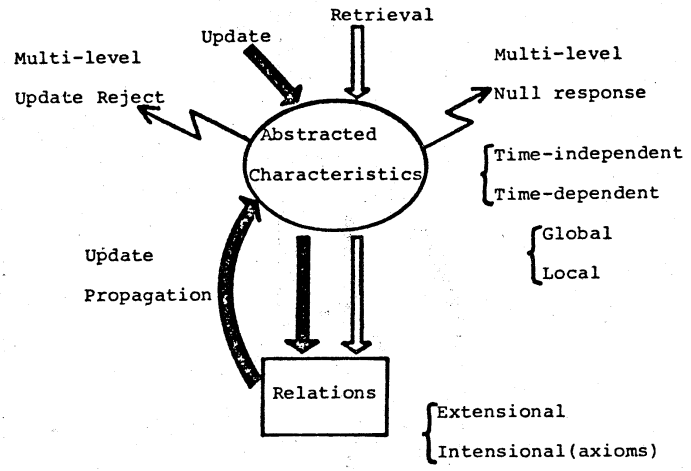


Fig.1(b) Abstracted Characteristics in Relational Databases

The major advantages of the usage of abstracted characteristics are as follows:

(1) Efficient processing of retrieval/update operations is possible,

(2) View updating mechanism can be made more powerful than conventional ones[CHAMG7505], [DAYAB7809],

(3) A high level user interface can be directly supported.

As shown in Fig.1(a), integrity and security constraints qualify the users' requests, e.g. retrieval/update operations to the database. In Fig.1(b), abstracted characteristics are used for providing a high level user interface, which can perform Yes/No and numerical

quantification queries efficiently and can provide some rough meaning of the corresponding answers without any reference to the relation files. Especially, when the answer is "Null" (or "No") its meaning can be provided according to the type of abstracted characteristics used in the conclusion. The major difference in this approach from the one in Joshi et al. [JOSHK77] is that this approach is independent of the order of the sub-queries to be processed. Moreover, concerning processing efficiency, we mention that the concept of functional dependency can be applied to processing of retrieval/update operations as well as to schema design. However, users' requests usually concern a small part of the database, while integrity constraints such as functional dependencies(FD's) deal with all tuples in base relations. We will show that not only FD's but also time-dependent local, time-dependent global FD's defined as abstracted characteristics in this paper play an important role in improving the processing efficiency.

In addition, given abstracted characteristics, we provide a powerful multi-level procedure for checking view update operations.

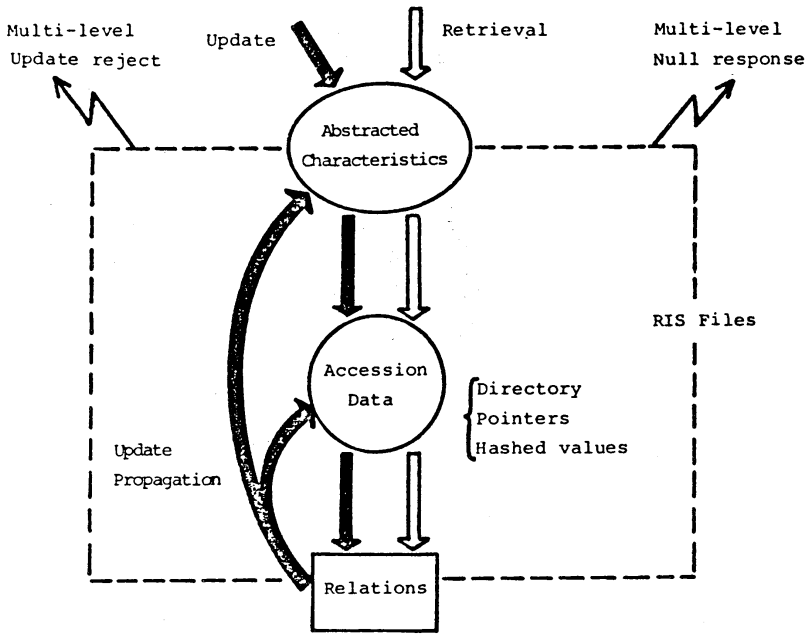


Fig.1(c) Abstracted Characteristics in Combination with RIS Files

Fig.1(c) describes an implementation of the above concept of abstracted characteristics with the support of a relational access method called Relational Inverted Structure(RIS) [TANAL7808].

2. Basic Concepts

This section is designed to deal with definitions of new concepts which are used in this paper. Terms concerning relational data model, e.g. functional dependencies, restriction, join, division, etc., can be referred in [CODD7105],[DATE77].

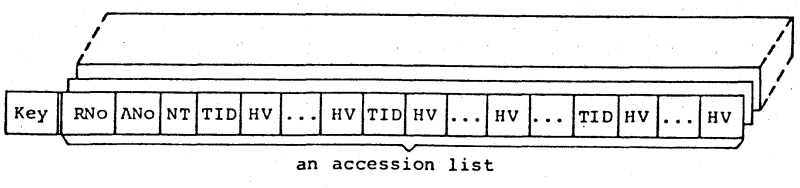
In the relational data model, a relation is represented by a table, which has n columns corresponding to n attributes, and many rows, each of which represents one n-tuple. A domain associated to an attribute is the set of all possible values of the attribute. The range of an attribute is defined for each combination of an attribute and a relation to be the set of all values appearing on the attribute in the relation. In order to distinguish the possible set of tuples which may appear in a relation and the set of tuples at an instance t in the relation, we call the former a relation schema and the latter the snap-shot or the extension at t of the relation schema.

A relation which is stored explicitly in a database is called a base relation. In this paper, a view is a virtual relation and is defined on base relations using relational operations such as projection, restriction, join and division. The update operations are defined as changing some attribute values, insertion, deletion of a single or a set of tuples.

Given two attributes A_i and A_j, the domain of A_i and A_j are denoted by $D(A_i)$ and $D(A_j)$, respectively. The term domain-related is defined as follows: (1) If $D(A_i) \cap D(A_j) \neq \emptyset$ then A_i and A_j are domain-related, denoted by $A_i \sim A_j$, (2) For distinct i, j and k, if $A_i \sim A_j$ and $A_j \sim A_k$, then $A_i \sim A_k$.

For a given domain-related attribute $A_i = \{A_{i1}, \ldots, A_{ik}\}$, a RIS file denoted by $RIS(A_i)$ can be defined [TANAL7808]. Hereafter, the term "domain value" x in $RIS(A_i)$ means a value which belongs to at least one $D(A_{ij})(j=1, \ldots, k)$.

A record according to a domain value in a RIS file consists of the domain value and a number of accession lists. Each accession list corresponds with a relation and an attribute, and has a list of tuple identification codes(TID's) whose corresponding tuples own the domain value. Incorporated into each TID is a list of hashed values of attributes other than the attribute A_{ij} in the relation. Fig.2 illustrates the structure of a RIS file record.



HV : Hashed value RNo : Relation number

NT : Number of tuples ANo : Attribute number

Fig.1: Structure of a RIS file record.

3. Query Processing Using Abstracted Characteristics

Abstracted characteristics wil be shown in this section to be useful for supporting relational operations efficiently.

Abstracted characteristics of data are defined as the information abstracted from sets of tuples in relation(s). A characteristic is global if it concerns all tuples in a snap-shot of a relation schema and is satisfied by all the tuples. A characteristic is local if it concerns only a subset of tuples in a snap-shot of a relation schema and is satisfied by all the tuples in the subset. A classification of abstracted characteristics is presented in [LE-K7902]. In this section, a brief description of global characteristics is given.

Global characteristics are classified into two types: (A) Time-independent global characteristics (TIGC) and (B) Time-dependent global characteristics (TDGC). A global characteristic is time-independent if it is satisfied by all snap-shots of the corresponding relation schema. A global characteristic is time-dependent if it is satisfied by some snap-shots of the relation schema but not all.

Some of TIGC's which correspond to integrity constraints are (a) Functional dependencies (FD's) and (b) Attribute ranges. The range of an attribute is defined to be the set of all values appearing in the attribute. The following TDGC's are very important for efficient processing of relational operations: (a) time-dependent functional dependencies(TDFD's) and (b) bound numbers of associated attribute values. Attribute set Y is said to be(time-dependent) functional dependent on attribute set X with respect to R^t, denoted by $X \xrightarrow{R^t} Y$ if and only if the values of Y are uniquely determined by the values of X in R^t. The bound numbers of Y values associated to X values are defined by the maximum and the minimum numbers of Y values associated to a X value.

3.1 Efficient Query Processing with Abstracted Characteristics

In this section, application of global characteristics to efficient query processing is explored. Two types of relational operations are considered: join and division operations.

Join operations: Consider the join operation J: R1[A=B]R2 which join R1 on A with R2 on B. In order to carry out J, the files of R1 and R2, or their indices must be searched sequentially. For simplicity, we suppose that types of A and B are both integer, and their ranges can be provided as global characteristics in the form of (min_A, max_A) and (min_B, max_B). From the definition of join operation, only attribute values belonging to the intersection of the two ranges(intervals) need be searched. Thus, we can evaluate the intersection of the two intervals and instead of searching all attribute values, we need only search the ones in the intersection. This method can be applied to n-ary joins as well as in the case of 2-ary join explained above.

Given a set of attribute ranges(intervals), their intersection and union can be calculated using algorithms in [WONGE7709]. Moreover, in order to efficiently perform a sequence of join operations, join planning can be considered [KAMB7901].

Division operations: we discuss the processing of division operations through the following example. Consider relations R1($B,\overline{B}$) and R2($C,\overline{C}$). For purposes of discussion, R1, and the projection of R2 on C are called the dividend relation and the divisor relation, respectively. Following the definition of division, a value of R1.$\overline{B}$ is a result value if the set of R1.B values corresponding to the R1.$\overline{B}$ value contains the set of R2.C values. If all R1.$\overline{B}$ values do not satisfy this condition, the result is an empty set. The response in this case is a negative one. Such an unsuccessful search can be sped up in the following case. If R1.$\overline{B}\rightarrow$R1.B holds for the dividend relation, the set of R1.B values corresponding to each R1.$\overline{B}$ value has only one element. Provided this functional dependency, if the set of R2.C values has more than one element, the result is empty. Therefore, a division with the dividend relation where the functional dependency R1.$\overline{B}\rightarrow$R1.B holds can be efficiently performed by first checking the number of values in the divisor relation. If the number is greater than one, no more redundant operations are needed, and the division results in an empty set.

The method described above can be extended for use with not only FD's but also with other abstracted characteristics such as TDFD's, bound numbers of attributes. If the TDFD R1.$\overline{B} \xrightarrow{R^t}$ R1.B holds at the time division is carried out, it can be used in the place of the FD R1.$\overline{B}\rightarrow$R1.B if such an FD does not exist. If the abstracted characteristic concerning $\overline{B}$ says that the maximum number of B values associated to each $\overline{B}$ value is n and the number of different values in C is k, and k>n, we can conclude that the response in this case is also a negative one. Note that in the above example, n=1.

3.2 Access Support Using Abstracted Characteristics and RIS Files

A RIS file is a generalized inverted file which stores: (1) Accession data, e.g. domain values, TID's(pointers), hashed values, etc, and (2)Abstracted characteristics, e.g. ranges of attributes, bound numbers, integrity constraints, etc.

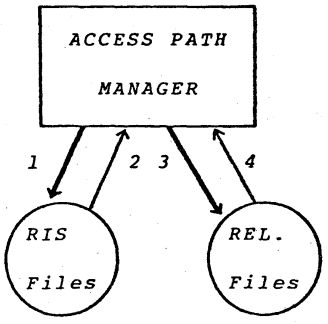


Fig.3 File Access Sequence Using RIS Files.

Fig.3 models the access sequence in a database system with the support of RIS files. Generally, an access sequence consists of the following steps: (1) the access path manager generates access commands to RIS files, (2) TID's, hashed values and abstracted characteristics stored in corresponding RIS files are obtained, (3) Using TID's the

access path manager gives access commands to relation files, and (4) obtains tuples(data) from these files. Using data obtained in step 2, redundant accesses to relation files are obmitted. Abstracted characteristics are used at the same time to increase the efficiency of processing. Access sequences for join and division operations are explained below.

Join operations: Consider the join operation J in Section 3.1. The RIS file for A and B is denoted by RIS(A). J is performed in the following steps:

Step 1: (Use of abstracted characteristics to reduce the search scope) Obtain the ranges of attributes A and B. Calculate their intersection.

Step 2: (Search RIS file for the addresses of resulting tuples) Search records of values belonging to the intersection range. In doing this, records containing TID's of only one relation, either R1 or R2 are omitted. Only records containing TID's of both R1 and R2 are obtained.

Step 3: (Read resulting tuples from relation files using TID's obtained in step 2) Relatio files of R1 and R2 are accessed directly through TID's obtained in step 2. Resulting tuples are obtained and concetenated for producing the results. Stop.

Division operations: Consider relation R1(A,$\bar{A}$) and R2(B,$\bar{B}$) and division operation Q:R1[A÷B]R2. Q is performed in the following steps:

Step 1: (Use of abstracted characteristics to check for an unsuccessful division) Obtain the number k of B values in relation R2. If k>n, where n is the maximum number of A values associated to each $\bar{A}$ value, then the result is empty. Stop. Otherwise, go to the next step.

Step 2: (Search RIS file for accession data) RIS(A) is sequentially accessed to obtain a set of (TID,h($\bar{A}$)) pairs of relation R1, where h($\bar{A}$) is the hashed value of the corresponding $\bar{A}$ value.

Step 3: (Check for unsuccessful search with accession data) If there is a record in RIS(A) which contains only TID's of R2, the division results in an empty set. Stop. Otherwise, go to the next step.

Step 4: (Select out the results and check) Each set of (TID,h($\bar{A}$)) obtained in step 2 corresponds to each B value in relation R2. Therefore, select out the TID's with h($\bar{A}$) appearing in all the sets. The TID's give the division results. Here, because hashed values are used instead of the values themselves, final check for hash collision must be carried out.

4. Checking the View Update Viability Using Abstracted Characteristics

Our goal in this section is to present the application aspect of abstracted characteristics of the view update problem. The view update problem was first pointed out by Codd[CODD7408]. Chamberlin et al.[CHAMG7505] has shown rules for a view update operation to be carried out without side effects. Here, we call such an update operation a viable operation. Related

works have also been presented in [TANAL7808],[KAMB7901], [DAYAB7809], etc.

In this section, first, conditions for checking the update viability of a view are given. Next, a checking procedure based on these conditions is discussed. In the procedure, RIS files which store abstracted characteristics are used to produce relevant checking data.

Let Bi and Vi ($1 <= i <= n$) with attribute sets $XiUYi$, and $X'iUY'i$, respectively be base relations and views. Views Vi is defined as follows: $V1=B1$, $V_{i+1}(X'_{i+1},Y'_{i+1})=B_{i+1}[Y_{i+1}=Y'_i]V_i$, for $1<=i<=n-1$. Here, only conditions for deletion operations are presented. For conditions for insertion operations, see [LE-K7902].

(A) Condition at the global level: Conditions similar with Proposition 1 given below are presented independently by Dayal,U. et al.[DAYAB7809] and Tanaka,K. et al.[TANAL7808].

Proposition 1: The sufficient condition for the deletion of arbitrary tuple t from view Vn to be viable is that either $Y_{i+1} \rightarrow X_{i+1}$ or $Y'_i \rightarrow X'_i$ or both hold on V_{i+1}, for $1<=i<=n-1$.

(B) Conditions at the local level:

Proposition 2: The deletion of tuple (x,y,z) from view V2 is viable if and only if there exists no tuple (x',y,z'), where $x' \neq x$, $x' \in V2[X'_1]$, $z' \neq z$, $z' \in V2[X2]$.

Proposition 3: The deletion of tuple t from view Vn is viable if and only if for every Vi, $2<= i<= n$, there exists no tuple (x,y,x'), where $x \in Vn[Xi]$, $x \neq t[Xi]$, $y=t[Yi]$, $y \in Vn[Yi]$, $y \in Vn[Y'_{i-1}]$, $x' \neq t[X'_{i-1}]$, $x' \in Vn[X'_{i-1}]$.

Checking Procedure: Using RIS files, a checking procedure are performed at the following levels provided that only update operations satisfying integrity and security constraints defined on the corresponding view are considered. RIS files which store abstracted characteristics will be used as inverted files, they provide relevant data for checking.

(1) Checking with time-independent global characteristics: Propositions at the global level are used to check whether or not the update operation can be viable. If by these Propositions, it is shown that the view can be updated by any update operations, they are carried out without checking. If not, every time an update operation is given, it must be checked by the steps below.

(2) Checking with time-dependent global characteristics: If instead of time-independent FD's, there exist corresponding time-dependent FD's, Propositions at the global level are used to check in the same way with step (1). However, because TDFD's may change at any instance, the view must be checked when updated if some of the TDFD's have changed. On the other hand, here the bound numbers associated with every join attribute can be used for checking update inviability.

For example, in V2(X1,Y1,X2) if the bound numbers of X1 and X2 associated with Y1 are equal or greater than 2, then no deletion can be update viable.

The following step is needed for other cases.

(3) Checking with local characteristics: At this level, local characteristics, mainly tuple numbers and associated hashed value numbers are used. Corresponding RIS files are directly accessed through given domain values(tuple values) and numbers of associated TID's and hashed

values are obtained. Checking is carried out following Propositions at the local level using the above data.

5. Support of High Level User Interface by Abstracted Characteristics

Abstracted characteristics are useful not only for efficient processing of retrieval/update operations, but also for supporting high level user interface. They can provide the following facilities:

(1) Yes/No queries are efficiently handled. Furthermore, abstracted characteristics are useful to let a user know the meaning of the results, especially 'No' answer without any references to relation files.

(2) Numerical quantifications such as 'at least n','exactly n' and 'at most n' queries are also efficiently handled.

(3) Domain-oriented queries are efficiently handled, such as queries to handle domain-values which do not appear in any relation.

Yes/No Queries: It is important to support Yes/No queries efficiently as well as queries to retrieve a set of tuples. Abstracted characteristics can be used to decrease the number of accesses to relation files in order to answer Yes/No queries. Global characteristics are mainly used to achieve a rapid unsuccessful search. After examining global characteristics, local characteristics are next used to examine the existence of the tuples with the specified value. Only when the existence is not determined, relation files are accessed to obtain the answer. When the answer is proved to be 'No', the meaning of "No' is usually different. That is, according to the type of abstracted characteristics which are used to derive "No', the meaning is different.

For example, let us consider the query: "Is there an employee whose managers are JOHN and SMITH?" If the FD EMP(employee) → MGR(manager) holds, then 'No' is derived on examining time-independent global characteristics. It means that the query is not consistent with the semantics of the database. In this manner, abstracted characteristics can partly provide the meaning of the answer for Yes/No queries.

Numerical Quantifications: Since bound numbers of associated attribute values are stored in several types of abstracted characteristics, numerical quantifications can be efficiently handled. For example, let us consider the query: "Find employees whose manager is SMITH and who has at least two children." Global characteristics can be used to achieve a rapid unsuccessful search for these types of queries. Then, local characteristics, concerned with bound numbers, can be used to obtain the answer. In this query, if the number of children of employees whose manager is 'SMITH' is also stored as local characteristics, the answer can be obtained.

Recently, Lacroix et al. introduced a high level user language called ILL[LACRP7710], in which numerical quantifications such as 'at least n' and 'at most n' are directly expressed as

AT LEAST n and AT MOST n, respectively. Abstracted characteristics are considered to be useful to support these kinds of relational database languages. Bound numbers stored as global/local characteristics can be used to achieve a rapid unsuccessful search for queries with numerical quantifications. Hashed values of attribute values associated with a specified value can be used to know directly 'at most' numbers.

Domain-oriented Queries: Lacroix et al. also pointed out that conventional tuple-oriented relational calculus is not sufficient to handle database domains[LACRP7710]. Especially, it cannot handle domain values that do not appear in any relation at a certain time. The domain-oriented relational calculus suggested by Lacroix and Pirotte, such as ILL, can directly interact with a database domain.

As described in Section 3, astracted characteristics are collected and managed for each database domain. Therefore, it seems to be able to handle efficiently queries concerned with domain-values.

6. Conclusions

A new concept concerning abstracted characteristics is presented and discussed for improving the processing efficiency of users' requests, providing a new powerful view update checking facility, and a high level user interface.

We intend to develop our works as follows: the class of views described in this paper has to be expanded to a general view class defined by axioms as being done in an inferential relational system[MINK7803]. The application of abstracted characteristics with the support of a relational access method such as RIS files to such an inferential relational system is believed to be useful for heuristic searches for responses as well as improvement of efficiency.

Acknowledgements The authors are grateful to the colleagues in Yajima Laboratory, especially to Mr. Masamitsu Yamamura for their cooperation. This work is partly supported by the Science Foundation Grant of the Ministry of Education, Science and Culture of Japan.

References: [CHAMG7505] CHAMBERLIN,D.D. et al.,Proc. AFIPS 1975 NCC, May 1975, [CODD7408] CODD,E.F., Proc. IFIP Congress 1974, [CODD7105] CODD.E.F., Courant Compt. Science Symposia, May 1971, [DATE77] DATE,C.J., Addison Wesley 1977, [DAYAB7809], DAYAL,U. et al. Proc. 4th International Conference on VLDB Sept. 1978, [JOSHK77] JOSHI,K.A. et al. Proc. 5th IJCAI 1977, [KAMB7901] KAMBAYASHI,Y. Yajima Lab. Res. Rep. ER79-02, Jan. 1979, [LACRP7710] LACROIX,M. et al. Proc. 3rd VLDB Oct. 1977, [MINK7803] MINKER,J., ACM Trans. DB Systems Vol.3, No.1, 1978, [WONGE7709] WONG,K.C., ACM Trans DB Systems, Vol.2, No.3, Sept. 1977, [TANAL7808] TANAKA,K. et al. Proc. International Conf. on Math. Studies of Infor. Processing, Aug. 1978, [LE-K7902] LE VIET,C. et al. Yajima Res. Rep. ER79-03, Feb. 1979.