Exhibit 22

Bell Communications Research

Date: December 31, 1987

From: J. W. Falk
51400/LCC 2E-311
6100

Subject: "A Fast Method for Storing and Retrieving Automatically Expiring Data - Job 693"

R.B.Robrock:

In response to your memorandum of December 2, 1987 to me, we have concluded that sufficient patentable novelty and commercial value resides in the above suggestion to warrant the filing of a patent application. This application, to be designated R. M. Nemes Case 2, is being prepared and will be forwarded to the inventor for his approval in due course.

J. W. Falk

jwf-ron

Copy to
P. D. Bloom
E. R. Byrne
B. N. Dickman
B. Edwards
A. R. Ephrath
B. F. Gardner, Jr.
R. M. Nemes
J. T. Peoples

Bell
Communications
Research

Date: December 2, 1987

Subject: **Request For Patentability Study**

From: **R. B. Robrock**
25600/RRC 4A473
(201) 699-2770

J. W. Falk

Jim:

This is to request that a patentability study be undertaken on the attached description of "A Fast Method for Storing and Retrieving Automatically Expiring Data" by R. Nemes of my organization. It is somewhat related to Patent Job 564 ("A Hybrid Hashing Technique") and ought to be handled in a similar way.

If you have any questions, please call me or R. Nemes (201) 699-4365.

R. B. Robrock

Copy (w/Att.) to
Division Managers 256
P. D. Bloom
E. R. Byrne
B. N. Dickman
B. Edwards
A. R. Ephrath
B. F. Gardner, Jr.
R. M. Nemes

Bell
Communications
Research

Date: September 9, 1987

From: R. Nemes
25655/RRC 4A373
(201) 699 – 4365

Subject: **Proposed Patent Application: A Fast Method for Storing and Retrieving Automatically Expiring Data**

Technical Memorandum

1. Introduction

Many fast techniques for storing and retrieving permanent data have long been known and used. Among them, hashing is often the preferred choice when storage space is deemed cheap relative to retrieval time, and, in principle, hashing can be considered as a tradeoff of space for time.

Temporary data, on the other hand, poses special problems, especially when the expiration of data items is not signaled explicitly. This can occur, for example, with data items that automatically expire after a certain length of time. One technique for dealing with data expiration in a hash table is to simply consider expired items "deleted," in the classic hashing sense of the term. But such a solution can lead to performance penalties when data lifetimes are short, because of deleted-entry *contamination*[1]. Another method for dealing with expired data is to periodically execute an off-line garbage collection procedure that scans the entire data base, cleaning up expired items using an algorithm similar to the one shown in Knuth [2] for deleted-entry garbage collection associated with linear probing under open addressing. This, however, is an unattractive solution because it requires taking the data base off line while the cleanup procedure runs, which can be a lengthy process.

The solution proposed here is an on-the-fly method (i.e., does not require taking the data base off line) that does incremental garbage collection in the neighborhood of a table slot that is hashed to. It has been successfully deployed in Bellcore's Line Information Data Base (LIDB) fraud monitoring subsystem, and is now being proposed for use in the transaction context save/restore facility of the Service Switching Node (SSN).

We begin by describing the classic technique of data storage and retrieval in a hash table.

---

1. A hash table is *contaminated* when an excessive number of entries are marked "deleted." Contamination lengthens data retrieval time.
2. D. Knuth, *The Art of Computer Programming, Volume 3, Sorting and Searching*, pg. 527, Addison-Wesley, Reading, Mass., 1973.

2. Classic Hashing

Taken as a whole, a hash table is a logically contiguous, circular list of consecutively numbered fixed-size storage units, called *cells*, each capable of storing a single item called a *record*. Each record contains a distinguished field, called the *key*, which is used as the basis for storing and retrieving the associated record. The keys throughout the data base are distinct. The mathematical function (or map) that associates a cell number with a key is called a *hashing function*. Hashing functions are usually not one-to-one in that they map many distinct keys to the same cell number.

2.1 Record Storage

To store a new record, a cell number is generated by invoking the hashing function on the key. If not occupied, the record is stored there. Otherwise, a *collision* has occurred and a cell must be found elsewhere, in an overflow area, using an appropriate collision resolution technique. The method employed here is known as *linear probing* under *open addressing*. Open addressing means that the overflow area is the hash table itself, and linear probing indicates sequential scanning of cells beginning with the next cell. (Recall that the table is viewed circularly.) Thus collisions are resolved by storing the record in the first unoccupied cell found.

2.2 Record Retrieval

Retrieving a record is similar. Invoking the hashing function on the search key yields a cell number. If the record is not found there, searching continues following the same path as record storage. An empty terminates an unsuccessful search.

2.3 Record Deletion Using "Deleted" Status

This technique for deleting a record requires distinct cell status indications of "empty," "occupied," and "deleted." Initially, all cells are marked "empty." When a record is stored in a cell, its status is changed to "occupied." The significance of the cell status is that an empty cell signals termination to the retrieval procedure, while a deleted cell indicates that the search must continue; either type of cell can store a new record.

To delete a record from the data base, the record is first located using the retrieval procedure described above. If found, the status of the vacated cell is changed from "occupied" to "deleted." Note that this deletion procedure, which is faster than Knuth's algorithm (shown below), can result in hash table contamination.

2.4 Knuth's Record Deletion Algorithm

Knuth's scheme uses only two cell markings, "empty" and "occupied." When deleting a record, the vacated cell's status is changed from "occupied" to "empty." Then, the chain[3] is traversed from that cell forward, searching for a record whose key hashes at or behind the vacated cell. If one is found, it is copied to the vacated cell, whose marking is changed from "empty" to "occupied." The procedure then repeats, now deleting the copied record from its original position. The end of the chain signals termination. Shown here is a recursive characterization of the algorithm.

3. A *chain* is defined as a consecutive sequence of occupied cells.

```
procedure knuth_delete (i: 0 .. table_size − 1);
    /* Delete cell i from hash table */
  procedure recursive_delete (j, k: 0 .. table_size − 1);
      /* Delete cell k instead of cell j if required */
    begin /* recursive_delete */
      if cell k is marked empty
        then mark cell j empty
        else if record in cell k hashes at or before position j
              then begin
                contents(j) := contents(k);
                recursive_delete (k, (k + 1) mod table_size)
              end /* then */
              else recursive_delete (j, (k + 1) mod table_size)
    end; /* recursive_delete */

  begin /* knuth_delete */
    recursive_delete (i, (i + 1) mod table_size)
  end /* knuth_delete */
```

3. Hashing Automatically Expiring Data

We now describe a hashing scheme based on linear probing under open addressing that possesses the following properties:

a. The table can never become contaminated.

b. The data base need never be taken off line.

c. An expired record is allowed to die in place. It is formally removed from the table using Knuth's algorithm whenever an insertion, retrieval, or deletion takes place in the neighborhood of that record.

d. The size of the neighborhood that is cleaned up each time an insertion, retrieval, or deletion is done can be controlled by specifying a maximum table load factor[4] and not accepting new insertions when

that maximum has been attained. Once the maximum has been reached, new insertions will be allowed only after cell cleanup done in association with retrievals, deletions, and attempted insertions brings the load factor down below the specified maximum.

3.1 Deleting Expired Entries On-The-Fly

We now describe how expired entries are removed from the table.

Each cell is always in one of two states: "empty" or "occupied." All cells are initially in the empty state. When a retrieval, deletion, or attempted insertion is made, the hashing function is invoked on the search key. If the target cell is part of a chain of occupied cells, then the entire chain is scanned sequentially (in reverse) and expired records are deleted using Knuth's algorithm.

Two additional computations are piggy-backed on the reverse scan of the target chain: the search key is compared with each unexpired record in the chain and, for an unsuccessful search, the appropriate empty cell to receive the new record is located.

3.2 Functions Provided

The following functions are made available to the application program:

insert (*record*: *record_type*)

Returns *replaced* if a record associated with *record.key* was found in the table and subsequently replaced.

Returns *inserted* if a record associated with *record.key* was not found in the table and the passed record was subsequently inserted.

Returns *full* if a record associated with *record.key* was not found in the table and passed record could not be inserted because load factor has reached *max_load_factor*.

retrieve (*record*: *record_type*)

Returns *success* if record associated with *record.key* was found in the table and assigned to *record*.

Returns *failure* if search was unsuccessful.

delete (*record_key*: *record_key_type*)

Returns *success* if record associated with *record_key* was found in the table and subsequently deleted.

Returns *failure* if none found.

4. The *load factor* is defined as the fraction of cells that are not marked "empty."

3.3 Logical Structure of Hash Table

The following formal definitions are required for specifying the insertion, retrieval, and deletion algorithms:

a. const *table_size* /* size of hash table */

b. const *max_load_factor* /* $0 \leq$ *max_load_factor* < 1 */

c. var *table*: array[0 .. *table_size* − 1] of *record_type*; /* hash table */

d. var *load*: 0 .. *table_size* − 1; /* number of occupied entries of hash table array (initially 0) */

3.4 Algorithms

Algorithms for the functions described above are now shown.

```
function insert (record: record_type): (replaced, inserted, full );

  var position: 0 .. table_size − 1;        /* position in table to update or insert (returned by search_table) */

  begin

    if search_table (record.key, position)

      then begin

        table[position] := record;
        return (replaced)

      end

      else if load/table_size < max_load_factor

        then begin

          load := load + 1;
          table[position] := record;
          return (inserted)

        end

        else return (full)

  end  /* insert */
```

```
function retrieve (var record: record_type): (success, failure);

 var position: 0 .. table_size − 1;        /* position in table where record resides (returned by search_table) */

 begin

  if search_table (record.key, position)

     then begin

       record := table[position];
       return (success)

     end

     else return (failure)

 end  /* retrieve */
```

```
function delete (record_key: record_key_type): (success, failure);

  var position: 0 .. table_size − 1;        /* position in table where record resides (returned by search_table) */
      dummy_variable: 0 .. table_size − 1;  /* last two arguments to knuth_delete are not relevant */

  begin

    if search_table (record_key, position)

      then begin

        knuth_delete (position, true, dummy_variable, dummy_variable);
        return (success)

      end

      else return (failure)

  end   /* delete */
```

```
function search_table (record_key: record_key_type; var position: 0 .. table_size − 1): boolean;

 /* search table for record_key and delete expired records in target chain;
    position is set to index of found record or appropriate empty cell */

 var i: 0 .. table_size − 1;      /* used for scanning chain, both forwards & backwards */
     pos_empty: 0 .. table_size − 1;      /* index of leftmost empty cell to right of position */

     is_rec_found: boolean;              /* indicates whether search is successful */


 begin

  position := hash (record_key);
  is_rec_found := false;

  if table[position] is not empty then

     begin

      i := position;      /* loop initialization */

      repeat             /* scan forward to end of chain containing table[position]  */

         i := (i + 1) mod table_size

      until (table[i] is empty);

      pos_empty := i;
      i := (i − 1 + table_size) mod table_size;

      while (table[i] is not empty) do      /*scan chain in reverse, deleting expired entries */

         begin

           if table[i] is expired then knuth_delete (i, is_rec_found,  position, pos_empty)

                                  else if table[i].key = record_key

                                                then begin

                                                  is_rec_found := true;
                                                  position := i

                                                end;

          i := (i − 1 + table_size) mod table_size

         end;   /* while */

      if not is_rec_found then position := pos_empty
```

```
    end;   /* then */
  return (is_rec_found)
end  /* search_table */
```

```
procedure knuth_delete (cell_to_del: 0 .. table_size − 1; is_rec_found: boolean;
                        var pos_of_search_rec, pos_empty: 0 .. table_size − 1);

 /* Delete table[cell_to_del] */

 var i, j: 0 .. table_size − 1;

 begin

   load := load − 1;

   do forever

     table[cell_to_del] := empty;

     if not is_rec_found then
       if (pos_of_search_rec ≤ cell_to_del < pos_empty)
         or (cell_to_del < pos_empty < pos_of_search_rec)
         or (pos_empty < pos_of_search_rec ≤ cell_to_del) then pos_empty := cell_to_del;

     i := cell_to_del;          /* save position of emptied slot */

     repeat          /* scan forward looking for a record to fill hole in chain */

       cell_to_del := (cell_to_del + 1) mod table_size;
       if table[cell_to_del] is empty then return;
       j := hash (table[cell_to_del].key)

     until (j ≤ i < cell_to_del) or (i < cell_to_del < j) or (cell_to_del < j ≤ i);

     table[i] := table[cell_to_del];          /* use table[cell_to_del] to plug hole in chain */

     if (is_rec_found) and (pos_of_search_rec = cell_to_del) then pos_of_search_rec := i

   end

 end   /* knuth_delete */
```

4. Concept Originality

The idea presented above is an original extension of table hashing that provides local on-the-fly incremental garbage collection of automatically expiring data.

R. Nemes
Member of Technical Staff
Evolutionary Services Development District

CONTENTS

Bell
Communications
Research

Date: **February 23, 1988**

Subject: **R. M. Nemes Case 2**

From: **Nancy A. Montesano**
51400/LCC 2E-347
740-6414

Mr. R. M. Nemes:

Attached for your files is a copy of the above-identified patent application, which was filed in the United States Patent and Trademark Office (USPTO) on February 2, 1988.

Until this application issues as a patent, these papers should be treated as Bellcore restricted information and should be safeguarded accordingly.

You will be apprised of the proceedings in the USPTO as they are disclosed to us.

Nancy A. Montesano
nkf

Nancy A. Montesano
Staff Manager
Intellectual Property Matters

NAM-nkf

Att.
As stated

Bell Communications Research

Date: **February 23, 1988**

Subject: **Patent Application Filed**

From: **Nancy A. Montesano**
51400/LCC 2E-347
740-6414

Mr. Paul D. Bloom:

For your information, enclosed is a copy of a patent application recently filed in the United States Patent and Trademark Office for a member of your division. Until this application issues as a patent, it should be treated as Bellcore restricted information and safeguarded accordingly.

If I can be of further assistance, please contact me on 740-6414.

Nancy A. Montesano nkf

Nancy A. Montesano
Staff Manager
Intellectual Property Matters

NAM-nkf

Att.
R. M. Nemes Case 2

C 693

Bell Communications Research

Date: April 21, 1988

Subject: Foreign Filing Recommendation
R. M. Nemes Case 2

From: R. O. Nimtz
51400/LCC 2E-303
377-4309

Mr. J. W. Falk:

The subject matter of this case relates to the removal of expired records in large data bases accessed by means of a hashing technique. More specifically, it deals with the removal of automatically expiring records. This invention is used in the Line Identification Data Base (LIDB) of the Service Control Point (SCP) System.

Although patent coverage for software inventions presents special prosecution problems in some countries, it is noted that sales of the SCP System are pending or completed in Italy, New Zealand and Taiwan. It is recommended that foreign filing be undertaken in these and any other countries in which SCP sales are contemplated.

R. O. Nimtz
Consultant

RON-ron

6/16/88

Volume 3 / **Sorting and Searching**

THE ART OF COMPUTER PROGRAMMING

Reading, Massachusetts
Menlo Park, California · London · Amsterdam · Don Mills, Ontario · Sydney

DONALD E. KNUTH *Stanford University*

ADDISON-WESLEY PUBLISHING COMPANY

Volume 3 / **Sorting and Searching**

THE ART OF COMPUTER PROGRAMMING

Reading, Massachusetts
Menlo Park, California · London · Amsterdam · Don Mills, Ontario · Sydney

DONALD E. KNUTH *Stanford University*

ADDISON-WESLEY PUBLISHING COMPANY

UNITED STATES DEPARTMENT OF COMMERCE
Patent and Trademark Office

Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231

SERIAL NUMBER	FILING DATE	FIRST NAMED APPLICANT	ATTORNEY DOCKET NO.
[illegible]	[illegible]	[illegible]	

JAMES W. FALK
BELL COMMUNICATIONS RESEARCH, INC.
290 WEST MOUNT PLEASANT AVE.
LIVINGSTON, NJ 07039

EXAMINER	
KULIK, P	
ART UNIT	PAPER NUMBER
237	9

DATE MAILED: 10/23/89

Below is a communication from the EXAMINER in charge of this application
COMMISSIONER OF PATENTS AND TRADEMARKS

ADVISORY ACTION

☒ THE PERIOD FOR RESPONSE:

☐ is extended to run _____ from the date of the Final Rejection

☐ continues to run _____ from the date of the Final Rejection

☒ expires three months from the date of the final rejection or as of the mailing date of this Advisory Action, whichever is later. In no event however, will the statutory period for response expire later than six months from the date of the final rejection.

Any extension of time must be obtained by filing a petition under 37 CFR 1.136(a), the proposed response and the appropriate fee. The date on which the response, the petition, and the fee have been filed is the date of the response and also the date for the purposes of determining the period of extension and the corresponding amount of the fee. Any extension fee pursuant to 37 CFR 1.17 will be calculated from the date that the shortened statutory period for response expires as set forth above.

☐ Appellant's Brief is due in accordance with 37 CFR 1.192(a).

☒ Applicant's response to the final rejection, filed _____, has been considered with the following affect, but it is not deemed to place the application in condition for allowance:

1. ☒ The proposed amendments to the claim and/or specification will not be entered and the final rejection stands because:
   a. ☐ There is no convincing showing under 37 CFR 1.116(b) why the proposed amendment is necessary and was not earlier presented.
   b. ☒ They raise new issues that would require further consideration and/or search. (See Note).
   c. ☐ They raise the issue of new matter. (See Note).
   d. ☐ They are not deemed to place the application in better form for appeal by materially reducing or simplifying the issues for appeal.
   e. ☐ They present additional claims without cancelling a corresponding number of finally rejected claims.

   NOTE: The addition of "means for identifying" and of the limitation of removing all expired records "each time said chain is accessed" is considered *

2. ☐ Newly proposed or amended claims _____ would be allowed if submitted in a separately filed amendment cancelling the non-allowable claims.

3. ☒ Upon the filing of an appeal, the proposed amendment ☐ will be ☐ will not be, entered and the status of the claims in this application would be as follows:

   Allowed claims: _____
   Claims objected to: _____
   Claims rejected: 1-10
   However;
   a. ☐ The rejection of claims _____ on references is deemed to be overcome by applicant's response.
   b. ☐ The rejection of claims _____ on non-reference grounds only is deemed to be overcome by applicant's response.

4. ☐ The affidavit, exhibit or request for reconsideration has been considered but does not overcome the rejection.

5. ☐ The affidavit or exhibit will not be considered because applicant has not shown good and sufficient reasons why it was not earlier presented.

☐ The propcsed drawing correction ☐ has ☐ has not been approved by the examiner.

☐ Other

* to substantially change the scope of the finally rejected claims and requires further consideration and search.

EDDIE P. CHAN
PRIMARY EXAMINER
ART UNIT 237

PTOL-303 (REV. 3-86)

IN THE UNITED STATES
PATENT AND TRADEMARK OFFICE

R. M. Nemes
Case 2

SERIAL NO. 07/151,639 FILED February 2, 1988

GROUP ART UNIT 237

EXAMINER Paul Kulik

TITLE Methods and Apparatus for Information Storage and Retrieval

THE COMMISSIONER OF PATENTS AND TRADEMARKS
WASHINGTON, D.C. 20231

SIR:

Enclosed is an amendment in the above-identified application. No additional fee is required, as shown below:

CLAIMS AS AMENDED						
(1)	(2) CLAIMS REMAINING AFTER AMENDMENT	(3)	(4) HIGHEST NUMBER PREV. PAID FOR	(5) PRES. EXTRA	(6) RATE	(7) ADDIT. FEE
TOTAL CLAIMS FOR FEE PURPOSES	8	MINUS	10	0	x $12	$0
INDEP. CLAIMS	2	MINUS	4	0	x $36	$0
MULTIPLE CLAIM(S) FIRST PRESENTED WITH THIS AMENDMENT		NO		IF YES + $120		$0
			TOTAL ADDITIONAL FEE FOR THIS AMENDMENT→			$0

In the event of any non-payment or improper payment of a required fee, the Commissioner is authorized to charge deposit account 02-1820 as required to correct the error.

Respectfully,

James W. Falk/jp
James W. Falk
Reg. No. 16154
Attorney for Applicant(s)

Date: OCT 10 1989

Bell Communications Research, Inc.
290 West Mount Pleasant Avenue, Room 2E-304
Livingston, New Jersey 07039

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Commissioner of Patents and Trademarks, Washington, D. C. 20231.

Pt. 16 OCT 10 1989

OCT 10 1989

Joann DeKanofski

IN THE UNITED STATES
PATENT AND TRADEMARK OFFICE

R. M. Nemes
Case No. 2
SERIAL NO. 07/151,639 FILED February 2, 1988
GROUP ART UNIT 237
EXAMINER Paul Kulik
TITLE Methods and Apparatus for Information Storage and Retrieval

THE COMMISSIONER OF PATENTS AND TRADEMARKS
WASHINGTON, D.C. 20231

SIR:

In response to the Final Office Action of August 4, 1989, (Paper No. 5), and substituting for the unentered amendment of August 31, 1989, please amend the above-identified application as follows:

IN THE SPECIFICATION:

Page 8, line 8, insert a period after "2".

IN THE CLAIMS:

Amend Claim 1 as follows:

1. (Twice Amended) An information storage and retrieval system using hashing techniques to provide rapid access to the records of said system and utilizing a linear probing technique to store records with the same hash address, at least some of said records automatically expiring [in response to the occurrence of an external event], said system comprising

a record search means utilizing a search key to access a chain of records having the same hash address,

said record search means including means for identifying and removing all expired ones of said records from said chain of records each time said chain is accessed, and

means, utilizing said record search means, for inserting, retrieving and deleting records from said system and, at the same time,

removing all expired ones of said records in the accessed chains of records.

Leave Claim 2 unamended as follows:

2. (Not Amended) The information storage and retrieval system according to claim 1 further comprising

means for recursively moving a record from a later position in said chain of records into the position of one of said expired records.

Amend Claim 3 as follows:

3. (Amended) The information storage and retrieval system according to claim 1 further including

means for counting the number of records in said system,

means , responsive to said counting means, for inhibiting the insertion of new records into said system when the number of records in said system exceeds [available storage space falls below] a preselected value.

Amend Claim 4 as follows:

4. (Twice Amended) The information storage and retrieval system according to claim 3 further including

means , also responsive to said counting means, for re-enabling the insertion of new records into said system when the number of records in said system falls below [available storage space rises above] said preselected value.

Cancel Claim 5.

Amend Claim 6 as follows:

6. (Twice Amended) A method for storing and retrieving information records using hashing techniques to provide rapid access to said records and utilizing a linear probing technique to store records with the same hash address, at least some of said records automatically expiring, said method comprising the steps of

accessing a chain of records having the same hash address,

identifying the automatically expired ones of said records,

[comparing the contents of each said record to at least one external event to determine which of said records has expired,]

removing all automatically expired records from said chain of records each time said chain is accessed, and

inserting, retrieving or deleting one of said records from said

13 system following said step of removing.

Leave Claim 7 unamended as follows:

1 7. (Not Amended) The method according to claim 6 further
2 comprising the step of

3 moving a record from a later position in said chain of records
4 into the position of one of said expired records.

Amend Claim 8 as follows:

1 8. (Twice Amended) The method according to claim 6
2 further comprising the steps [step] of

3 counting the number of records in said system, and

4 inhibiting the insertion of new records into said system when
5 the number of records in said system rises above [available storage space
6 falls below] a preselected value.

Amend Claim 9 as follows:

1 9. (Amended) The method according to claim 8 further
2 comprising the step of

3 re-enabling the insertion of new records into said system
4 when the number of records in said system falls below [available storage
5 space rises above] said preselected value.

Cancel Claim 10.

Remarks

All of the claims in the case, both amended claims and claims not amended, have been repeated in this amendment for the convenience of the Examiner and of applicants' attorney. The amended claims are so identified.

Claims 1, 3 and 4 and claims 6, 8 and 9 have been amended. Claims 5 and 10 have been canceled. Claims 1-4 and 6-9 therefore remain in the case.

Applicant's attorney, Robert O. Nimtz (Reg. No. 18645), would like to thank the Examiner in Art Unit 237 for the courtesies extended to him during a telephone interview on October 3, 1989. While no agreements were reached as to the allowability of any claim or the enterability of any amendment, nevertheless these discussions have been of assistance to Applicant's attorney in preparing this amendment.

The rejection of the Office Action of March 24, 1989, has been maintained, repeated and made final. The proposed amendment of August 31, 1989, was not entered because it raised new issues requiring further consideration or search, the Examiner commenting that "The addition of limitations involving 'self-expiring' records requires further consideration and search." In view of this comment, the offending language has been omitted from the present amendments. Since this language was added only for the purposes of clarification, and since the claims are believed to clearly distinguish over the prior art without these limitations, resubmittal of the amendment without the "self-expiring" language is deemed appropriate.

In the Final Rejection of August 4, 1989, claims 1, 2, 5, 6, 7 and 10 were rejected under 35 U.S.C. 102(b) as being anticipated by the D. E. Knuth text *The Art of Programming*, "Sorting and Searching," pages 506-549, Addison-Wesley Series in Computer Science and Information Processing, Reading, Massachusetts, 1973. As admitted at page 3, line 4, of applicant's specification, Knuth discloses a non-contaminating hash table deletion procedure which removes the record to be deleted and moves other records to close the search chain opened up by the removal of the deleted record. By refusing to distinguish between "expired" and "deleted" records, the Examiner has read applicants claims to the removal of expired records on Knuth's removal of deleted records. Applicant, however, is his own lexicographer and has defined "expired" at page 2, lines 14-17, as records which "have a limited lifetime after which they become obsolete." A "deleted" record, on the other hand, is one which is marked as deleted by a contaminating deletion procedure, or physically deleted from the hash table by a non-contaminating deletion procedure. Note that an expired record must still be deleted by either a contaminating or a non-contaminating deletion procedure.

Significantly, Claim 1 recites "at least some of said records automatically expiring." Clearly, Knuth teaches nothing concerning automatically expiring records. Claim 1, as amended, goes on to recite "means for identifying and removing all expired ones of said records from said chain of records each time said chain is accessed." Since Knuth teaches nothing about automatically expiring records, he also does not teach the identification and removal of such records. Knuth relies on the user of his deletion procedures to identify the records to be deleted. Moreover, records are removed in the Knuth system only by deletion procedures. Claim 1, however, calls for the removal of automatically expiring

records "each time said chain is accessed," whether for insertion, retrieval or user-initiated deletion. This "incremental garbage collection" (page 3, line 28, of specification) takes place automatically, with no requirement for the user to do anything but to use the system as if no such garbage collection were required. Claim 1, as amended, is believed to clearly distinguish over the Knuth reference. Applicant's claim 1 describes a system which elegantly solves a problem not even appreciated by Knuth. Hence applicant's solution can hardly be said to be obvious in view of Knuth.

Claim 2 is dependent on claim 1 and is believed to distinguish over Knuth for the same reasons as the parent claim. Claim 5 has been canceled without prejudice in favor of the claims retained.

Like claim 1, claim 6 has been amended to call for "at least some of said records being automatically expiring," "identifying automatically expired ones of said records" and "removing all automatically expired ones of said records from said chain of records each time said chain is accessed." Claim 6 is believed to distinguish over the Knuth reference for the same reasons as claim 1.

Claim 7 is dependent on claim 6 and is believed to distinguish over Knuth for the same reasons as the parent claim. Claim 10 has been canceled without prejudice in favor of the claims retained.

Claims 3, 4, 8 and 9 were rejected under 35 U.S.C. 103 as being unpatentable over the same Knuth reference, the Examiner commenting that it would have been obvious to those of ordinary skill in the art that "keeping the number of insertions to a minimum level would increase the performance" and hence it would have been obvious "to prohibit the number of insertions as claimed." The Examiner notes that these rejected claims do not "call for counting records of a table." This objection is well-taken and claim 3 has been amended to call for "means for counting the number of records in said system" and "means, responsive to said counting means, for inhibiting the insertion of new records into said system when the number of records in said system exceeds a preselected value." Clearly, Knuth does not teach this specific solution to the problem of over-loading a hash table. Furthermore, claim 3 is dependent on claim 1 and hence further distinguishes from Knuth in the same manner as the parent claim.

Claim 4 is dependent on claim 3 and is believed to distinguish over the Knuth reference in the same manner as the parent claim.

Claim 8, like claim 3, calls for "counting the number of records in said system" and "inhibiting the insertion of new records into said system when the number of records in said system rises above a preselected value." Claim 8 is thus believed to distinguish over Knuth for the same reasons as claim 3. Claim 9 is dependent on claim 8 and thus also distinguishes from Knuth for these same reasons.

In conclusion, it is believed that all of the claims remaining in the case, claims 1-4 and 6-9, are neither anticipated by nor obvious from the Knuth teachings. Applicant has recognized the problem of automatically expiring records and provided a solution to this problem. Knuth has not even recognized the problem, much less suggested applicant's solution to the problem. Indeed, it can be said that Knuth's failure to mention the problem tells the person of ordinary skill that no such problem exists, and hence Knuth teaches away from rather than toward applicant's invention.

It is believed that all of the claims in this case, as amended, clearly distinguish over the Knuth text and over all other art of record. Allowance and passage to issue are respectfully requested.

Respectfully submitted,
R. M. Nemes

By _____
James W. Falk, Attorney
Reg. No. 16154
(201) 740-6100

Bell Communications Research, Inc.
290 West Mount Pleasant Avenue, Room 2E-304
Livingston, New Jersey 07039

Date: OCT 10 1989

UNITED STATES DEPARTMENT OF COMMERCE
Patent and Trademark Office

Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231

SERIAL NUMBER	FILING DATE	FIRST NAMED APPLICANT	ATTORNEY DOCKET NO.
07/[illegible]	02/02/88	[illegible]	

JAMES H. FALK
BELL COMMUNICATIONS RESEARCH, INC.
290 WEST MOUNT PLEASANT AVE.
LIVINGSTON, NJ 07039

EXAMINER	
FULLER F	
ART UNIT	PAPER NUMBER
237	7

DATE MAILED: 09/14/89

Below is a communication from the EXAMINER in charge of this application
COMMISSIONER OF PATENTS AND TRADEMARKS

ADVISORY ACTION

☒ THE PERIOD FOR RESPONSE:

☐ is extended to run _____ from the date of the Final Rejection

☐ continues to run _____ from the date of the Final Rejection

☒ expires three months from the date of the final rejection or as of the mailing date of this Advisory Action, whichever is later. In no event however, will the statutory period for response expire later than six months from the date of the final rejection.

Any extension of time must be obtained by filing a petition under 37 CFR 1.136(a), the proposed response and the appropriate fee. The date on which the response, the petition, and the fee have been filed is the date of the response and also the date for the purposes of determining the period of extension and the corresponding amount of the fee. Any extension fee pursuant to 37 CFR 1.17 will be calculated from the date that the shortened statutory period for response expires as set forth above.

☐ Appellant's Brief is due in accordance with 37 CFR 1.192(a).

☒ Applicant's response to the final rejection, filed 8/31/89. has been considered with the following affect, but it is not deemed to place the application in condition for allowance:

1. ☒ The proposed amendments to the claim and/or specification will not be entered and the final rejection stands because:
   a. ☐ There is no convincing showing under 37 CFR 1.116(b) why the proposed amendment is necessary and was not earlier presented.
   b. ☒ They raise new issues that would require further consideration and/or search. (See Note).
   c. ☐ They raise the issue of new matter. (See Note).
   d. ☐ They are not deemed to place the application in better form for appeal by materially reducing or simplifying the issues for appeal.
   e. ☐ They present additional claims without cancelling a corresponding number of finally rejected claims.

   NOTE: The addition of limitations involving 'self-expiring' records requires further consideration and search.

2. ☐ Newly proposed or amended claims _____ would be allowed if submitted in a separately filed amendment cancelling the non-allowable claims.

3. ☒ Upon the filing of an appeal, the proposed amendment ☐ will be ☒ will not be, entered and the status of the claims in this application would be as follows:

   Allowed claims: _____
   Claims objected to: _____
   Claims rejected: 1-10

   However;
   a. ☐ The rejection of claims _____ on references is deemed to be overcome by applicant's response.
   b. ☐ The rejection of claims _____ on non-reference grounds only is deemed to be overcome by applicant's response.

4. ☐ The affidavit, exhibit or request for reconsideration has been considered but does not overcome the rejection.

5. ☐ The affidavit or exhibit will not be considered because applicant has not shown good and sufficient reasons why it was not earlier presented.

☐ The proposed drawing correction ☐ has ☐ has not been approved by the examiner.

☐ Other

GARETH D. SHAW
SUPERVISORY PATENT EXAMINER
ART UNIT 237

PTOL-303 (REV. 3-86)

C

IN THE UNITED STATES
PATENT AND TRADEMARK OFFICE

R. M. Nemes
Case 2

SERIAL NO. 07/151,639 FILED February 2, 1988

GROUP ART UNIT 237

EXAMINER Paul Kulik

TITLE Methods and Apparatus for Information Storage and Retrieval

THE COMMISSIONER OF PATENTS AND TRADEMARKS
WASHINGTON, D.C. 20231

SIR:

Enclosed is an amendment in the above-identified application. No additional fee is required, as shown below:

CLAIMS AS AMENDED						
(1)	(2) CLAIMS REMAINING AFTER AMENDMENT	(3)	(4) HIGHEST NUMBER PREV. PAID FOR	(5) PRES. EXTRA	(6) RATE	(7) ADDIT. FEE
TOTAL CLAIMS FOR FEE PURPOSES	8	MINUS	10	0	x $12	$0
INDEP. CLAIMS	2	MINUS	4	0	x $36	$0
MULTIPLE CLAIM(S) FIRST PRESENTED WITH THIS AMENDMENT		NO		IF YES + $120		$0
			TOTAL ADDITIONAL FEE FOR THIS AMENDMENT→			$0

In the event of any non-payment or improper payment of a required fee, the Commissioner is authorized to charge deposit account 02-1820 as required to correct the error.

Respectfully,

James W. Falk/jp

James W. Falk
Reg. No. 16154
Attorney for Applicant(s)

Date: AUG 2 9 1989

Bell Communications Research, Inc.
290 West Mount Pleasant Avenue, Room 2E-304
Livingston, New Jersey 07039

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Commissioner of Patents and Trademarks, Washington, D. C. 20231.

on Pt. 16 AUG 2 9 1989

Date AUG 2 9 1989

Joann Pekarofski
Joann Pekaroftski

B

IN THE UNITED STATES
PATENT AND TRADEMARK OFFICE

R. M. Nemes
Case No. 2
SERIAL NO. 07/151,639 FILED February 2, 1988
GROUP ART UNIT 237
EXAMINER Paul Kulik
TITLE Methods and Apparatus for Information Storage and Retrieval

This amendment not accepted by the PTO.

THE COMMISSIONER OF PATENTS AND TRADEMARKS
WASHINGTON, D.C. 20231

SIR:

In response to the Final Office Action of August 4, 1989, (Paper No. 5), please amend the above-identified application as follows:

IN THE SPECIFICATION:

Page 8, line 8, insert a period after "2".

IN THE CLAIMS:

Amend Claim 1 as follows:

1. (Twice Amended) An information storage and retrieval system using hashing techniques to provide rapid access to the records of said system and utilizing a linear probing technique to store records with the same hash address, at least some of said records being automatically self-expiring [expiring] in response to the contents of said self-expiring records [occurrence of an external event], said system comprising

a record search means utilizing a search key to access a chain of records having the same hash address,

said record search means including means for identifying and removing all expired ones of said self-expiring records from said chain of records each time said chain is accessed, and

means, utilizing said record search means, for inserting, retrieving and deleting records from said system and, at the same time, removing all expired ones of said self-expiring records in the accessed chains

of records.

Amend Claim 2 as follows:

2. (Amended) The information storage and retrieval system according to claim 1 further comprising

means for recursively moving a record from a later position in said chain of records into the position of one of said expired ones of said self-expiring [expired] records.

Amend Claim 3 as follows:

3. (Amended) The information storage and retrieval system according to claim 1 further including

means for counting the number of records in said system,

means , responsive to said counting means, for inhibiting the insertion of new records into said system when the number of records in said system exceeds [available storage space falls below] a preselected value.

Amend Claim 4 as follows:

4. (Twice Amended) The information storage and retrieval system according to claim 3 further including

means , also responsive to said counting means, for re-enabling the insertion of new records into said system when the number of records in said system falls below [available storage space rises above] said preselected value.

Cancel Claim 5.

Amend Claim 6 as follows:

6. (Twice Amended) A method for storing and retrieving information records using hashing techniques to provide rapid access to said records and utilizing a linear probing technique to store records with the same hash address, at least some of said records being self-expiring in response to the contents of said self-expiring records, said method comprising the steps of

accessing a chain of records having the same hash address,

identifying the expired ones of said self-expiring records,

[comparing the contents of each said record to at least one external event to determine which of said records has expired,]

removing all of the expired ones of said self-expiring [expired] records from said chain of records, and

inserting, retrieving or deleting one of said records from said system following said step of removing.

Amend Claim 7 as follows:

7. (Amended) The method according to claim 6 further comprising the step of

moving a record from a later position in said chain of records into the position of one of said removed self-expiring [expired] records.

Amend Claim 8 as follows:

8. (Twice Amended) The method according to claim 6 further comprising the steps [step] of

counting the number of records in said system, and

inhibiting the insertion of new records into said system when the number of records in said system rises above [available storage space falls below] a preselected value.

Amend Claim 9 as follows:

9. (Amended) The method according to claim 8 further comprising the step of

re-enabling the insertion of new records into said system when the number of records in said system falls below [available storage space rises above] said preselected value.

Cancel Claim 10.

Remarks

All of the claims in the case, both amended claims and claims not amended, have been repeated in this amendment for the convenience of the Examiner and of applicants' attorney. The amended claims are so identified.

Claims 1 through 4 and 6 through 9 have been amended. Claims 5 and 10 have been canceled. Claims 1-4 and 6-9 therefore remain in the case.

The rejection of the Office Action of March 24, 1989, has been maintained, repeated and made final. Claims 1, 2, 5, 6, 7 and 10 are rejected under 35 U.S.C. 102(b) as being anticipated by the D. E. Knuth text *The Art of Programming*, "Sorting and Searching," pages 506-549, Addison-Wesley Series in Computer Science and Information Processing, Reading, Massachusetts, 1973. As

admitted at page 3, line 4, of applicant's specification, Knuth discloses a non-contaminating hash table deletion procedure which removes the record to be deleted and moves other records to close the search chain opened up by the removal of the deleted record. By refusing to distinguish between "expired" and "deleted" records, the Examiner has read applicants claims to the removal of expired records on Knuth's removal of deleted records. Applicant, however, is his own lexicographer and has defined "expired" at page 2, lines 14-17, as records which "have a limited lifetime after which they become obsolete." A "deleted" record, on the other hand, is one which is marked as deleted by a contaminating deletion procedure, or physically deleted from the hash table by a non-contaminating deletion procedure. Note that an expired record must still be deleted by either a contaminating or a non-contaminating deletion procedure.

In order to make this distinction abundantly clear, Claim 1 has been amended to recite "records being automatically self-expiring in response to the contents of said self-expiring records." Clearly, Knuth teaches nothing concerning self-expiring records. Claim 1, as amended, goes on to recite "means for identifying and removing all expired ones of said self-expiring records from said chain of records each time said chain is accessed." Since Knuth teaches nothing about self-expiring records, he also does not teach the identification and removal of such records. Knuth relies on the user of his deletion procedures to identify the records to be deleted. Moreover, records are removed in the Knuth system only by deletion procedures. Claim 1 calls for the removal of self-expiring records "each time said chain is accessed," whether for insertion, retrieval or user-initiated deletion. This "incremental garbage collection" (page 3, line 28, of specification) takes place automatically, with no requirement for the user to do anything but to use the system as if no such garbage collection were required. Claim 1, as amended, is believed to clearly distinguish over the Knuth reference. Applicant's claim 1 describes a system which elegantly solves a problem not even appreciated by Knuth. Hence applicant's solution can hardly be said to be obvious in view of Knuth.

Claim 2 is dependent on claim 1 and is believed to distinguish over Knuth for the same reasons as the parent claim. Claim 5 has been canceled without prejudice in favor of the claims retained.

Like claim 1, claim 6 has been amended to call for "records being self-expiring in response to the contents of said self-expiring records," "identifying

the expired ones of said self-expiring records" and "removing all of the expired ones of said self-expiring records from said chain of records." Claim 6 is believed to distinguish over the Knuth reference for the same reasons as claim 1.

Claim 7 is dependent on claim 6 and is believed to distinguish over Knuth for the same reasons as the parent claim. Claim 10 has been canceled without prejudice in favor of the claims retained.

Claims 3, 4, 8 and 9 were rejected under 35 U.S.C. 103 as being unpatentable over the same Knuth reference, the Examiner commenting that it would have been obvious to those of ordinary skill in the art that "keeping the number of insertions to a minimum level would increase the performance" and hence it would have been obvious "to prohibit the number of insertions as claimed." The Examiner notes that these rejected claims do not "call for counting records of a table." This objection is well-taken and claim 3 has been amended to call for "means for counting the number of records in said system" and "means, responsive to said counting means, for inhibiting the insertion of new records into said system when the number of records in said system exceeds a preselected value." Clearly, Knuth does not teach this specific solution to the problem of over-loading a hash table. Furthermore, claim 3 is dependent on claim 1 and hence further distinguishes from Knuth in the same manner as the parent claim.

Claim 4 is dependent on claim 3 and is believed to distinguish over the Knuth reference in the same manner as the parent claim.

Claim 8, like claim 3, calls for "counting the number of records in said system" and "inhibiting the insertion of new records into said system when the number of records in said system rises above a preselected value." Claim 8 is thus believed to distinguish over Knuth for the same reasons as claim 3. Claim 9 is dependent on claim 8 and thus also distinguishes from Knuth for these same reasons.

In conclusion, it is believed that all of the claims remaining in the case, claims 1-4 and 6-9, are neither anticipated by nor obvious from the Knuth teachings. Applicant has recognized the problem of self-extinguishing records and provided a solution to this problem. Knuth has not even recognized the problem, much less suggested applicant's solution to the problem. Indeed, it can be said that Knuth's failure to mention the problem tells the person of ordinary skill that no such problem exists, and hence Knuth teaches away from rather than toward

applicant's invention.

It is believed that all of the claims in this case, as amended, clearly distinguish over the Knuth text and over all other art of record. Allowance and passage to issue are respectfully requested.

Respectfully submitted,
R. M. Nemes

By _____
James W. Falk, Attorney
Reg. No. 16154
(201) 740-6100

Bell Communications Research, Inc.
290 West Mount Pleasant Avenue, Room 2E-304
Livingston, New Jersey 07039

Date: AUG 2 9 1989

UNITED STATES DEPARTMENT OF COMMERCE
Patent and Trademark Office

Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231

SERIAL NUMBER	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.
07/151,639	02/02/88	NEMES	

JAMES W. FALK
BELL COMMUNICATIONS RESEARCH, INC.
290 WEST MOUNT PLEASANT AVE.
LIVINGSTON, NJ 07039

EXAMINER
KULIK, P

ART UNIT	PAPER NUMBER
237	5

DATE MAILED: 08/04/89

This is a communication from the examiner in charge of your application.
COMMISSIONER OF PATENTS AND TRADEMARKS

FINAL

☒ This application has been examined ☒ Responsive to communication filed on 5/11/87 ☒ This action is made final.

A shortened statutory period for response to this action is set to expire 3 month(s), ~~_____ days~~ from the date of this letter.
Failure to respond within the period for response will cause the application to become abandoned. 35 U.S.C. 133

Part I THE FOLLOWING ATTACHMENT(S) ARE PART OF THIS ACTION:

1. ☒ Notice of References Cited by Examiner, PTO-892.
2. ☐ Notice re Patent Drawing, PTO-948.
3. ☐ Notice of Art Cited by Applicant, PTO-1449.
4. ☐ Notice of Informal Patent Application, Form PTO-152.
5. ☐ Information on How to Effect Drawing Changes, PTO-1474.
6. ☐ _____

Part II SUMMARY OF ACTION

1. ☒ Claims 1-10 are pending in the application.
   Of the above, claims _____ are withdrawn from consideration.
2. ☐ Claims _____ have been cancelled.
3. ☐ Claims _____ are allowed.
4. ☒ Claims 1-10 are rejected.
5. ☐ Claims _____ are objected to.
6. ☐ Claims _____ are subject to restriction or election requirement.
7. ☐ This application has been filed with informal drawings under 37 C.F.R. 1.85 which are acceptable for examination purposes.
8. ☐ Formal drawings are required in response to this Office action.
9. ☐ The corrected or substitute drawings have been received on _____. Under 37 C.F.R. 1.84 these drawings are ☐ acceptable. ☐ not acceptable (see explanation or Notice re Patent Drawing, PTO-948).
10. ☐ The proposed additional or substitute sheet(s) of drawings, filed on _____ has (have) been ☐ approved by the examiner. ☐ disapproved by the examiner (see explanation).
11. ☐ The proposed drawing correction, filed on _____, has been ☐ approved. ☐ disapproved (see explanation).
12. ☐ Acknowledgment is made of the claim for priority under U.S.C. 119. The certified copy has ☐ been received ☐ not been received ☐ been filed in parent application, serial no. _____; filed on _____.
13. ☐ Since this application appears to be in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under Ex parte Quayle, 1935 C.D. 11; 453 O.G. 213.
14. ☐ Other

1, 6 & 8 amended in Amendment A

EXAMINER'S ACTION

PTOL-326 (Rev. 6-88)

1. This application has been examined in response to the amendment filed May 11, 1989. The amended claims and applicants remarks have been considered but the application is not in condition for allowance for the reasons noted below.

2. The rejection of the last office action is maintained and repeated below. The applicants remarks are addressed following the rejection.

3. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless-
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

4. Claims 1, 2, 5, 6, 7, and 10 are rejected under 35 U.S.C. 102(b) as being anticipated by "The art of Computer Programming", Sorting and Searching, D.E. Knuth, Addison-Wesley Sevies in Computer Science and Information Processing, pages 506-549, 1973.

5. Claim one is rejected over Knuth because Knuth discloses "means for removing all expired records from said chain of records" as indicated by the applicants on page 3 of the instant specification. Claim one does not distinguish over the prior art as discussed by the applicants. Unlike claim 5, claim 1 does not indicate that all expired records are removed each time records

are accessed. This claim therefore reads on the prior art procedure of taking database off-line to remove all expired records.

6. Claim 6 is rejected for the same reasons as was claim one.

7. Claims 2 and 7 are rejected because they also do not distinguish over Knuth. As noted by applicants the algorithm of Knuth moves records appearing in later position into positions of the expired records.

8. Regarding claims 5 and 10, the algorithm discussed in Knuth discloses the claimed invention. As noted in Knuth, Brent's variation, page 525, it is known to move records when inserting an item. As noted by the applicant's page 527 of Knuth discloses moving records when deleting an item.

9. Claims 1, 2, 5, 6, 7, and 10 have been rejected.

10. The following is a quotation of 35 U.S.C. 103 which forms the basis for all obviousness rejections set forth in this Office action:

> A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

> Subject matter developed by another person, which qualifies as prior art only under subsection (f) and (g) of section 102 of this title, shall not preclude patentability under this section where the subject matter and the claimed invention were, at the time the invention was made, owned by the same person or subject to an obligation of assignment to the same person.

11. Claims 3, 4, 8 and 9 are rejected under 35 U.S.C. 103 as being unpatentable over "The art of Computer Programming," D.E. Knuth, pages 506-549, 1973.

12. Claims 3 and 4 recite prohibiting insertion of records when available storage space falls below a preselected value and re-enabling insertion of records when available storage space rises above this value. Claims 8 and 9 include the same limitations.

13. Knuth discusses available storage space of a table size M (see pages 519-521) and teaches that the linear probing algorithm works fine until the table begins to get full (i.e., the number of insertions increases). Knuth teaches that as the number of table insertions (N) increases (as N approaches M), the performance of linear probing degrades. With this in mind, it would have been obvious to those of ordinary skill in the art at the time of the invention that keeping the number of insertions to a minimum level would increase the performance of linear probing because this is what Knuth is clearly suggesting. It therefore would have been obvious in order to increase system performance to prohibit the number of insertions as claimed.

14. Claims 1-10 have been rejected.

15. The applicants remark that the examiner fails to distinguish between expired and deleted records and that, as a result, the rejection is untenable. The examiner does not agree however because applicants distinction between expired and deleted records is not fully related in the claims. That is to say - a deleted record may be an expired record.

16. It is clear that a response in the prior art to the expiration of records has been to mark them as deleted. Considering this, claims 1 and 6 fail to distinguish over the prior art practice of marking unwanted (expired) records as deleted and then removing those records from a chain of records. The Applicant remarks regarding claim 6 and Knuth, that Knuth clearly does not disclose or suggest "comparing the contents of each said record to at least one external event to determine which of said records has expired". The examiner notes that the prior art practice of deleting expired records clearly requires this step as someone or something must determine which records are to be deleted. The applicants arguments regarding claims 1, 2, 6 and 10 are therefore not considered persuasive.

17. Regarding claims 5 and 10, applicants argue that the algorithm Brent's Variation is not relevant to the automatic removal of expired records. While this may be true, the examiner relies on Knuth as disclosing the removal of deleted (and hence expired) records when a table is accessed. See for instance page 526 of Knuth

- second paragraph in the discussion of deletions. Applicants arguments in this respect are therefore not considered persuasive.

18. With respect to claims 3, 4, 8 and 9, applicant remarks that Knuth's recognition of the problem being solved by the invention of these claims is not tantamount to the suggestion of applicants solution and that from Knuth it is not obvious "to count the number of records in the table and to inhibit insertion when this number exceeds a threshold". To this extent, applicant implies the use of hindsight by the examiner. The examiner notes that the claimed invention does not call for counting records of a table but only a means for inhibiting and a means for reenabling insertion of records. The examiner maintains that such means are obvious over Knuth because it is clear from Knuth that if one wants to keep the 'performance' of linear probing from degrading then one must keep the number of insertions in a table from increasing. The obvious way to do this is to stop inserting records into the table. Applicants arguments regarding claims 3, 4, 8 and 9 are therefore not persuasive.

19. THIS ACTION IS MADE FINAL. Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a). The practice of automatically extending the shortened statutory period an additional month upon the filing of a timely first response to a final rejection has been discontinued by the Office. See 1021 TMOG 35.

A SHORTENED STATUTORY PERIOD FOR RESPONSE TO THIS FINAL ACTION IS SET TO EXPIRE THREE MONTHS FROM THE DATE OF THIS ACTION. IN THE EVENT A FIRST RESPONSE IS FILED WITHIN TWO MONTHS OF THE MAILING DATE OF THIS FINAL ACTION AND THE ADVISORY ACTION IS NOT MAILED UNTIL AFTER THE END OF THE THREE-MONTH SHORTENED STATUTORY PERIOD, THEN THE SHORTENED STATUTORY PERIOD WILL EXPIRE ON THE DATE THE ADVISORY ACTION IS MAILED, AND ANY EXTENSION FEE PURSUANT TO 37 CFR 1.136(a) WILL BE CALCULATED FROM THE MAILING DATE OF THE ADVISORY ACTION. IN NO EVENT WILL THE STATUTORY PERIOD FOR RESPONSE EXPIRE LATER THAN SIX MONTHS FROM THE DATE OF THIS FINAL ACTION.

20. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Paul Kulik whose telephone number is (703) 557-4999.

Any inquiry of a general nature or relating to the status of this application should be directed to the Group receptionist whose telephone number is (703) 557-2878.

P.K.
PK/jrm

8/1/89

GARETH D. SHAW
SUPERVISORY PATENT EXAMINER
ART UNIT 237

FORM PTO-892 (REV. 3-78) U.S. DEPARTMENT OF COMMERCE PATENT AND TRADEMARK OFFICE	SERIAL NO.	GROUP ART UNIT	ATTACHMENT TO PAPER NUMBER
NOTICE OF REFERENCES CITED	151639	237	5
	APPLICANT(S) Nemes		

U.S. PATENT DOCUMENTS

*		DOCUMENT NO.	DATE	NAME	CLASS	SUB-CLASS	FILING DATE IF APPROPRIATE
	A						
	B						
	C						
	D						
	E						
	F						
	G						
	H						
	I						
	J						
	K						

FOREIGN PATENT DOCUMENTS

*		DOCUMENT NO.	DATE	COUNTRY	NAME	CLASS	SUB-CLASS	PERTINENT SHTS. DWG	PERTINENT PP. SPEC.
	L								
	M								
	N								
	O								
	P								
	Q								

OTHER REFERENCES (Including Author, Title, Date, Pertinent Pages, Etc.)

*	R	"The Art of Computer Programming," Sorting and Searching, D.E. Knuth, Addison-Wesley Series in Computer Science and Information Processing, pgs. 506-549, 1973.
	S	
	T	
	U	

RECEIVED
Bell Communications Research, Inc.
AUG 7 1989

EXAMINER	DATE
[signature]	7/27/89

* A copy of this reference is not being furnished with this office action.
(See Manual of Patent Examining Procedure, section 707.05 (a).)

IN THE UNITED STATES
PATENT AND TRADEMARK OFFICE

R. M. Nemes
Case 2

SERIAL NO. 07/151,639 FILED February 2, 1988

GROUP ART UNIT 237

EXAMINER Paul Kulik

TITLE Methods and Apparatus for Information Storage and Retrieval

THE COMMISSIONER OF PATENTS AND TRADEMARKS
WASHINGTON, D.C. 20231

SIR:

Enclosed is an amendment in the above-identified application. No additional fee is required, as shown below:

CLAIMS AS AMENDED						
(1)	(2) CLAIMS REMAINING AFTER AMENDMENT	(3)	(4) HIGHEST NUMBER PREV. PAID FOR	(5) PRES. EXTRA	(6) RATE	(7) ADDIT. FEE
TOTAL CLAIMS FOR FEE PURPOSES	10	MINUS	10	0	x $12	$0
INDEP. CLAIMS	4	MINUS	4	0	x $36	$0
MULTIPLE CLAIM(S) FIRST PRESENTED WITH THIS AMENDMENT		NO		IF YES + $120		$0
			TOTAL ADDITIONAL FEE FOR THIS AMENDMENT→			$0

In the event of any non-payment or improper payment of a required fee, the Commissioner is authorized to charge deposit account 02-1820 as required to correct the error.

Respectfully,

James W. Falk/jp

James W. Falk
Reg. No. 16154
Attorney for Applicant

Date: MAY 9 1989

Bell Communications Research, Inc.
290 West Mount Pleasant Avenue, Room 2E-304
Livingston, New Jersey 07039

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Commissioner of Patents and Trademarks, Washington, D. C. 20231.
on MAY 9 1989
Date MAY 9 1989

Joann DeKarofski
Joann DeKarofski

IN THE UNITED STATES
PATENT AND TRADEMARK OFFICE

R. M. Nemes

Case 2

SERIAL NO. 07/151,639 FILED February 2, 1988

GROUP ART UNIT 237

EXAMINER Paul Kulik

TITLE Method and Apparatus for Information Storage and Retrieval

THE COMMISSIONER OF PATENTS AND TRADEMARKS
WASHINGTON, D.C. 20231

SIR:

In response to the Office Action of March 24, 1989 (Paper No. 3), please amend the above-identified application as follows:

IN THE SPECIFICATION:

Page 8, line 7, insert a period after "FIG. 2".

IN THE CLAIMS:

Amend Claim 1 as follows:

1. An information storage and retrieval system using hashing techniques to provide rapid access to the records of said system and utilizing a linear probing technique to store records with the same hash address, at least some of said records automatically expiring in response to the occurrence of an external event, said system comprising

a record search means utilizing a search key to access a chain of records having the same hash address,

said record search means including means for removing all expired ones of said records from said chain of records, and

means, utilizing said record search means, for inserting, retrieving and deleting records from said system.

Leave Claim 2 unamended as follows:

2. The information storage and retrieval system according to claim 1 further comprising

means for recursively moving a record from a later position in said chain of records into the position of one of said expired records.

Leave Claim 3 unamended as follows:

3. The information storage and retrieval system according to claim 1 further including

means for inhibiting the insertion of new records into said system when the available storage space falls below a preselected value.

Amend Claim 4 as follows:

4. The information storage and retrieval system ~~according~~ to claim 3 [wherein] further including

~~means for re-enabling~~ the insertion of new records into said ~~system~~ when the available storage space rises above said preselected value.

Leave Claim 5 unamended as follows:

5. An automatically decontaminating hashed storage table comprising

means for accessing said storage table for inserting, retrieving and deleting records, and

means for automatically removing expired records from said table each time said table is accessed.

Amend Claim 6 as follows:

6. A method for storing and retrieving information records using hashing techniques to provide rapid access to said records and utilizing a linear probing technique to store records with the same hash address, said method [system] comprising the steps of

accessing a chain of records having the same hash address,

comparing the contents of each said record to at least one external event to determine which of said records has expired,

removing all expired records from said chain of records, and

[utilizing said record search means, for] inserting, retrieving or [and] deleting one of said records from said system following said step of removing.

Leave Claim 7 unamended as follows:

7. The method according to claim 6 further comprising the step of

moving a record from a later position in said chain of records into the position of one of said expired records.

Amend Claim 8 as follows:

8. The method according to claim 6 further ~~comprising~~ [including] the step of

~~inhibiting~~ the insertion of new records into said system when ~~the available~~ storage space falls below a preselected value.

Leave Claim 9 unamended as follows:

9. The method according to claim 8 further comprising the step of

re-enabling the insertion of new records into said system when the available storage space rises above said preselected value.

Leave Claim 10 unamended as follows:

10. A method for automatically decontaminating a hashed storage table comprising the steps of

accessing said storage table for inserting, retrieving and deleting records, and

automatically removing expired records from said table each time said table is accessed.

Remarks

The specification has been corrected at page 8. Claims 1, 4, 6 and 8 have been amended. All of the claims remaining in the case, both amended claims and claims not amended, have been repeated in this amendment for the convenience of the Examiner and of applicants' attorney. The amended claims are so identified.

Claims 6 through 9 were rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention. More particularly, with regard to claim 6, the Examiner notes that "said record search means" has no antecedent and that claim 6 recites a method even though the introduction recites "system." The Examiner is entirely correct in this criticism and claim 6 is herewith amended to correct the deficiencies alluded to. Claim 6 now recites a "method" and calls for "inserting, retrieving or deleting one of said records

from said system following said step of removing." It is believed that claim 6, as amended, fully meets the requirement of 35 U.S.C. 112. Claims 7 through 9, dependent on claim 6, were rejected as being dependent on and including the deficiencies of claim 6. These claims are now dependent on a parent claim believed to fully satisfy the requirements of 35 U.S.C. 112, second paragraph.

Claims 1, 2, 5, 6, 7 and 10 were rejected under 35 U.S.C. 102(b) as being anticipated by the text *The Art of Computer Programming*, Donald E. Knuth, Addison-Wesley Publishing Company, 1973, pages 506-549. As discussed in the introduction to applicant's specification, page 1, lines 18-31, the Knuth text provides basic background material concerning hash storage techniques. Knuth does disclose, at page 527 of his text, and as noted by applicant at page 3, line 5, a technique for removing all *deleted* records. Knuth teaches nothing concerning *expired* records, as defined by applicant at page 2, lines 14-17 of the specification. As is readily apparent, the need for removing deleted records is similar to the need for removing expired records. Records become expired, however, without any action on the part of the user, and hence there is no "deletion activity" initiated by the user to delete the record, either by a contaminating algorithm or by a non-contaminating algorithm. It is for this reason that applicant detects *expired* records on each access to the data base. Thus, even if the data base is accessed for the purpose of deleting a particular record (FIG. 7 and pseudo-code at page 19 of the specification), the access routine looks for and removes *expired* records before removing the record to be deleted. The Examiner's failure to distinguish between expired records and records marked as deleted has led to an extremely inaccurate characterization of the Knuth reference and an obviously unsupported rejection of applicant's claims.

Claim 1, for example, as amended, calls for "means for removing expired ones of said records." Knuth teaches nothing about records expiring due to time lapse or the occurrence of an external event. In Knuth, the only way to delete a record is for the user to execute one of the delete algorithms that Knuth teaches. However, in order to make this difference clearer, claim 1 has been amended to recite "at least some of said records automatically expiring in response to the occurrence of an external event." Knuth clearly does not even teach this problem, much less applicant's solution to this problem. The problem that Knuth is addressing is the contamination that results from marking records as deleted without actually removing them from the data base. A non-contaminating deletion such as Knuth's can be used to prevent contamination, even with applicant's

invention. So too could a faster, contaminating deletion algorithm be used with applicant's invention. Applicant's invention simply is not concerned with the deletion of records by the user, but with removal of expired records independent of any explicit deletion action initiated by the user. Applicant has taken advantage of normal accesses of the record chains, whether for the purpose of retrieval, insertion or deletion, in order to examine the records of the chain for expiration, and if expired, to remove them.

Claims 2 is dependent on claim 1 and is believed to distinguish over the Knuth reference for the same reasons as claim 1.

Claim 5 recites "means for automatically removing *expired* records from said table each time said table is accessed" (emphasis added). As noted above, Knuth teaches nothing about *expired* records, much less automatic removal of such expired records. The Examiner points to the so-called "Brent's Variation" at page 525 of the Knuth reference. Brent's Variation is a variation of a record insertion algorithm which uses double hashing. In a double hashing system, if the original hash address is occupied, the algorithm uses an entirely different hashing function on the same key to obtain an offset from the original hash address. The new record is inserted here or, if it is filled, at new addresses offset by the same increment. Brent's Variation of this algorithm moves the record at the original hash address to its own offset address to make room for the new record. Clearly, Brent's Variation teaches nothing whatsoever about expiring records nor about removing expired records by closing the chain at a single hash address. Claim 5 is therefore also believed to clearly distinguish over the Knuth reference.

Claim 6, as amended, calls for "comparing the contents of each said record to at least one external event to determine which of said records has expired" and "removing all expired records." Knuth teaches nothing concerning expired records and does not teach or suggest comparing the contents of the records to external events. Claim 6, as amended, is believed to clearly distinguish over the Knuth reference.

Claims 7 through 9 are dependent on claim 6 and are believed to distinguish over the Knuth reference for the same reasons.

Claim 10, like claim 5, calls for "automatically removing expired records from said table each time said table is accessed." Again, Knuth teaches nothing concerning expired records, much less automatic removal of such expired

records.

It is believed that claims 1, 2, 5, 6, 7 and 10 clearly distinguish over the Knuth reference of record for the reasons noted above. Moreover, it is believed that the inventions of these claims are not obvious in view of the Knuth reference inasmuch as Knuth does not even disclose the expiring record problem, much less suggest applicant's solution to this problem.

Claims 3, 4, 8 and 9 were rejected under 35 U.S.C. 103 as being unpatentable over the same Knuth reference. These claims recite the technique of inhibiting insertions of new records when the available storage space falls below a preselected value, and re-enabling insertions when the available space rises above the preselected value. Claims 3, 4, 8 and 9 are dependent on claims 1 and 6 and are believed to distinguish over the Knuth reference in the same manner as their respective parent claims. Moreover, the Examiner's citation to Knuth's recognition of the problem (pp. 519-521) that applicant solves is *not* tantamount to the suggestion of applicant's solution. From Knuth's recognition that a problem exists, it is *not* obvious to count the number of records in the table and to inhibit insertion when this number exceeds a threshold. This is using applicant's own disclosure as a portion of the prior art in order to label a distinct contribution as obvious. The determination of obviousness, however, must be made based on the art available at the time of applicant's filing date, and not on applicant's disclosure.

The remaining references, cited but not applied by the Examiner, have been carefully reviewed and are not believed to be sufficiently pertinent to warrant detailed consideration here.

It is believed that all of the claims in this case, as amended, clearly distinguish over the Knuth reference and over all other art of record. Allowance and passage to issue are respectfully solicited.

Respectfully submitted,
R. M. Nemes

By _____
James W. Falk, Attorney
Reg. No. 16154
(201) 740-6100

Bell Communications Research, Inc.
290 West Mount Pleasant Avenue, Room 2E-304
Livingston, New Jersey 07039

Date: MAY 9 1989

UNITED STATES DEPARTMENT OF COMMERCE
Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231

SERIAL NUMBER	FILING DATE	FIRST NAMED APPLICANT	ATTORNEY DOCKET NO.
07/[illegible]609	02/02/89	NEMES	

JAMES W. FALK
BELL COMMUNICATIONS RESEARCH , INC.
290 WEST MOUNT PLEASANT AVE.
LIVINGSTON, NJ 07039

BLANK EXAMINER	
ART UNIT 257	PAPER NUMBER 3

DATE MAILED: 03/24/89

This is a communication from the examiner in charge of your application.

COMMISSIONER OF PATENTS AND TRADEMARKS

☒ This application has been examined ☐ Responsive to communication filed on _____ ☐ This action is made final.

A shortened statutory period for response to this action is set to expire 3 month(s), _____ days from the date of this letter.
Failure to respond within the period for response will cause the application to become abandoned. 35 U.S.C. 133

Part I THE FOLLOWING ATTACHMENT(S) ARE PART OF THIS ACTION:

1. ☒ Notice of References Cited by Examiner, PTO-892.
2. ☐ Notice re Patent Drawing, PTO-948.
3. ☒ Notice of Art Cited by Applicant, PTO-1449
4. ☐ Notice of informal Patent Application, Form PTO-152
5. ☐ Information on How to Effect Drawing Changes, PTO-1474
6. ☐ _____

Part II SUMMARY OF ACTION

1. ☒ Claims 1-10 are pending in the application.

   Of the above, claims _____ are withdrawn from consideration.

2. ☐ Claims _____ have been cancelled.

3. ☐ Claims _____ are allowed.

4. ☒ Claims 1-10 are rejected.

5. ☐ Claims _____ are objected to.

6. ☐ Claims _____ are subject to restriction or election requirement.

7. ☐ This application has been filed with informal drawings which are acceptable for examination purposes until such time as allowable subject matter is indicated.

8. ☐ Allowable subject matter having been indicated, formal drawings are required in response to this Office action.

9. ☐ The corrected or substitute drawings have been received on _____. These drawings are ☐ acceptable; ☐ not acceptable (see explanation).

10. ☐ The ☐ proposed drawing correction and/or the ☐ proposed additional or substitute sheet(s) of drawings, filed on _____, has (have) been ☐ approved by the examiner. ☐ disapproved by the examiner (see explanation).

11. ☐ The proposed drawing correction, filed _____, has been ☐ approved. ☐ disapproved (see explanation). However, the Patent and Trademark Office no longer makes drawing changes. It is now applicant's responsibility to ensure that the drawings are corrected. Corrections MUST be effected in accordance with the instructions set forth on the attached letter "INFORMATION ON HOW TO EFFECT DRAWING CHANGES", PTO-1474.

12. ☐ Acknowledgment is made of the claim for priority under 35 U.S.C. 119. The certified copy has ☐ been received ☐ not been received ☐ been filed in parent application, serial no. _____; filed on _____.

13. ☐ Since this application appears to be in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under Ex parte Quayle, 1935 C.D. 11; 453 O.G. 213.

14. ☐ Other

PTOL-326 (Rev. 7-82) EXAMINER'S ACTION

(REV. 7-80) 3-83 U.S. DEPARTMENT OF COMMERCE PATENT AND TRADEMARK OFFICE

INFORMATION DISCLOSURE STATEMENT

(Use several sheets if necessary)

MAY 1988 U.S. PATENT & TRADEMARK OFFICE

CASE NO.	SERIAL NO.
2	151,639
APPLIC: Richard M. Nemes	
FILING DATE: February 2, 1988	GROUP: 231

U.S. PATENT DOCUMENTS

*EXAMINER INITIAL		DOCUMENT NUMBER	DATE	NAME	CLASS	SUBCLASS	FILING DATE IF APPROP.
	AA						
	AB						
	AC						
	AD						
	AE						
	AF						
	AG						
	AH						
	AI						
	AJ						
	AK						

FOREIGN PATENT DOCUMENTS

		DOCUMENT NUMBER	DATE	COUNTRY	CLASS	SUBCLASS	TRANSLAT. YES
	AL						
	AM						
	AN						
	AO						
	AP						

I Hereby certify that this correspondence is being deposited with the United States Postal Service as first class mailed in an envelope addressed to: Commissioner of Patents and Trademarks, Washington, D.C. 20231, on MAY 4 1988

Date MAY 4 1988 Wanda M. Kirkland Wanda M. Kirkland

OTHER (*Including Author, Title, Date, Pertinent Pages, Etc.*)

P.K.	AR	"The Art of Computer Programming", Sorting and Searching, D. E. Knuth, Addison-Wesley Series in Computer Science and Information Processing, pgs. 506-549, 1973.
P.K.	AS	"Data Structures with Abstract Data Types and Pascal", D. F. Stubbs and N. W. Webre, Brooks/Cole Publishing Company, 1985, Section 7.4, Hashed Implementations, pgs. 310-336.
P.K.	AT	"Data Structures and Program Design", R. L. Kruse, Prentice-Hall, Inc. 1984, Section 3.7, Hashing, pgs. 112-126.

EXAMINER Paul V. Kyle DATE CONSIDERED 2/28/89

*EXAMINER: Initial if considered

TO SEPARATE, FOLD TOP AND BOTTOM EDGES, SNAP–APART AND DISCARD CARBON

6

FORM PTO-892 (REV. 3-78) U.S. DEPARTMENT OF COMMERCE PATENT AND TRADEMARK OFFICE	SERIAL NO.	GROUP ART UNIT	ATTACHMENT TO PAPER NUMBER	
NOTICE OF REFERENCES CITED	151639	237		3
	APPLICANT(S) Nemes			

U.S. PATENT DOCUMENTS

*		DOCUMENT NO.	DATE	NAME	CLASS	SUB-CLASS	FILING DATE IF APPROPRIATE
✓	A	4775932	10/4/88	Oxley et al.	364	200	
✓	B	4711524	12/29/87	Oxley et al.	364	200	
✓	C	4502118	2/26/85	Hagenmaier, Jr. et al.	364	200	
✓	D	4447875	5/8/84	Bolton et al.	364	200	
✓	E	4215402	7/29/80	Mitchell et al.	364	200	
✓	F	4121286	10/17/78	Venton et al.	364	900X	
	G						
	H						
	I						
	J						
	K						

FOREIGN PATENT DOCUMENTS

*		DOCUMENT NO.	DATE	COUNTRY	NAME	CLASS	SUB-CLASS	PERTINENT SHTS. DWG	PERTINENT PP. SPEC.
	L								
	M								
	N								
	O								
	P								
	Q								

OTHER REFERENCES (Including Author, Title, Date, Pertinent Pages, Etc.)

R

S

T

U

EXAMINER	DATE
Paul V. Fyles	3/2/89

* A copy of this reference is not being furnished with this office action.
(See Manual of Patent Examining Procedure, section 707.05 (a).)

(REV. 7-80) 3-83 / PATENT AND TRADEMARK OFFICE

CASE NO. 2 | SERIAL NO. 151,639

INFORMATION DISCLOSURE STATEMENT

(Use several sheets if necessary)

APPLICANT Richard M. Nemes

FILING DATE February 2, 1988 | GROUP 231

U.S. PATENT DOCUMENTS

*EXAMINER INITIAL		DOCUMENT NUMBER	DATE	NAME	CLASS	SUBCLASS	FILING DAT IF APPROPRIA*
	AA						
	AB						
	AC						
	AD						
	AE						
	AF						
	AG						
	AH						
	AI						
	AJ						
	AK						

FOREIGN PATENT DOCUMENTS

		DOCUMENT NUMBER	DATE	COUNTRY	CLASS	SUBCLASS	TRANSLATIO YES	NO
	AL							
	AM							
	AN							
	AO							
	AP							

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Commissioner of Patents and Trademarks, Washington, D.C. 20231, on MAY 4 1988

Date MAY 4 1988 Wanda M. Kiebler for WANDA M. KIEBLER

OTHER *(Including Author, Title, Date, Pertinent Pages, Etc.)*

	AR	"The Art of Computer Programming", Sorting and Searching, D. E. Knuth, Addison-Wesley Series in Computer Science and Information Processing, pgs. 506-549, 1973.
	AS	"Data Structures with Abstract Data Types and Pascal", D. F. Stubbs and N. W. Webre, Brooks/Cole Publishing Company, 1985, Section 7.4, Hashed Implementations, pgs.310-336.
	AT	"Data Structures and Program Design", R. L. Kruse, Prentice-Hall, Inc. 1984,Section 3.7, Hashing, pgs.112-126.

EXAMINER | DATE CONSIDERED

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609; Draw line through citation if not in

The references are discussed in the specification.

Richard M. Nemes

By _____
James W. Falk, Attorney
Reg. No. 16154
201, 740-6100

Bell Communications Research, Inc.

Date: MAY 4 1988

Attached
References AR-AT

FILING RECEIPT

UNITED STATES DEPARTMENT OF COMMERCE
Patent and Trademark Office
ASSISTANT SECRETARY AND COMMISSIONER
OF PATENTS AND TRADEMARKS
Washington, D.C. 20231

SERIAL NUMBER	FILING DATE	GRP. ART UNIT	FIL FEE REC'D	ATTORNEY DOCKET NO.	DRWGS	TOT CL	IND CL
07/151,639	02/02/88	231	$ 374.00	2	6	10	4

JAMES W. FALK
BELL COMMUNICATIONS RESEARCH , INC.
290 WEST MOUNT PLEASANT AVE.
LIVINGSTON, NJ 07039

Receipt is acknowledged of the patent application identified herein. It will be considered in its order and you will be notified as to the examination thereof. Be sure to give the U.S. SERIAL NUMBER, DATE OF FILING, NAME OF APPLICANT, and TITLE OF INVENTION when inquiring about this application. Fees transmitted by check or draft are subject to collection. Please verify the accuracy of the data presented on this transmittal.

Applicant(s)

RICHARD M. NEMES, BROOKLYN, NY.

FOREIGN FILING LICENSE GRANTED 03/21/88
TITLE
METHODS AND APPARATUS FOR INFORMATION STORAGE AND RETRIEVAL

PRELIMINARY CLASS: 364

(see reverse)

Subclass

IN TE UNITED STATES PATENT AN TRADEMARK OFFICE

Richard Michael Nemes

CASE 2

TITLE Methods and Apparatus for Information Storage and Retrieval

THE COMMISSIONER OF PATENTS AND TRADEMARKS
WASHINGTON, D.C. 20231

SIR:

Enclosed are the following papers relating to the above named application for patent:

[X] 6 sheets of drawing(s) plus two additional copies
[X] Assignment to Bell Communications Research, Inc.
[X] Specification
[X] Declaration and Power of Attorney
[X] Associate Power of Attorney
[ ] Information Disclosure Statement
[ ] Other

CLAIMS AS FILED				
(1)	(2) NUMBER FILED	(3) NUMBER EXTRA	(4) RATE	(5) BASIC FEE $340
TOTAL CLAIMS FOR FEE PURPOSES	10 - 20 =	0	x $12	0
INDEPENDENT CLAIMS	4 - 3 =	1	x $34	34.00
CLAIMS FILED INCLUDE MULTIPLE DEPENDENT CLAIM(S)	NO [X]	YES [ ]	IF YES, +$110	0
RECORDING FEE(S)				$ 7.00
			TOTAL FEE	$ 381.00

A check is enclosed in the amount of $381.00 to cover the filing fee and the cost of recording the assignment(s).

In the event of any non-payment or improper payment of a required fee, the Commissioner is authorized to charge deposit account 02-1820 as required to correct the error.

Please file the application and record the assignment(s), returning the latter to me.

Respectfully,

James W. Falk /wmk

Attorney for Applicant(s)

Date: FEB 1 1988

Bell Communications Research, Inc.
290 West Mount Pleasant Avenue - Room 2E-304
Livingston, NJ 07039

PT. 13 8/87 (rdp)

METHODS AND APPARATUS FOR INFORMATION STORAGE AND RETRIEVAL

Technical Field

This invention relates to information storage and retrieval systems and, more particularly, to the use of hashing techniques in such systems.

Background of the Invention

Information or data stored in a computer-controlled storage mechanism can be retrieved by searching for a particular key in the stored records. The stored record with a key matching the search key is then retrieved. Such searching techniques require repeated accesses or probes into the storage mechanism to perform key comparisons. In large storage and retrieval systems, such searching, even if augmented by efficient search algorithms such as a binary search, often requires an excessive amount of time.

Another well-known and much faster method for storing and retrieving information from computer store involves the use of so-called "hashing" techniques. These techniques are also sometimes called scatter-storage or key-transformation techniques. In a system using hashing, the key is operated upon (by a hashing function) to produce a storage address in the storage space (called the hash table). This storage address is then used to access the desired storage location directly with fewer storage accesses or probes than sequential or binary searches. Hashing techniques are described in the classic text by D. Knuth entitled The Art of Computer Programming, Volume 3, Sorting and Searching, pp.506-549, Addison-Wesley, Reading, Massachusetts, 1973.

Hashing functions are designed to translate the universe of keys into addresses uniformly distributed throughout the hash table. Typical hashing operations include truncation, folding, transposition and modulo arithmetic. A disadvantage of hashing techniques is that

more than one key can translate into the same storage address, causing "collisions" in storage or retrieval operations. Some form of collision-resolution strategy (sometimes called "rehashing") must therefore be provided. For example, the simple strategy of searching forward from the initial storage address to the first empty storage location will resolve the collision. This latter technique is called linear probing. If the hash table is considered to be circular so that addresses beyond the end of the table map back to the beginning of the table, then the linear probing is done with "open addressing," i.e., with the entire hash table as overflow space in the event that a collision occurs.

Some forms of data records have a limited lifetime after which they become obsolete. Scheduling activities, for example, involves records which become obsolete after the scheduled activity has occurred. Such record storage locations cannot be simply emptied since this location may be a link in a chain of locations previously created during a collision-resolution procedure. The classic solution to this problem is to mark the record as "deleted" rather than as "empty," and to leave the record in place. In time, however, the storage space can become contaminated by an excessive number of deleted or obsolete storage locations that must be searched to locate desired records. With the passage of time, such storage contamination can reduce the performance of retrieval operations below acceptable levels. Problems of this type are discussed in considerable detail in Data Structures and Program Design, by R. L. Kruse, Prentice-Hall, Englewood Cliffs, New Jersey, 1984, pp. 112-126, and Data Structures with Abstract Data Types and PASCAL, by D. F. Stubbs and N. W. Webre, Brooks/Cole Publishing, Monterey, California, 1985, pp. 310-336.

In the prior art, such storage space contamination was avoided by deletion procedures that

eliminated deleted records by replacing the deleted record with another record in the collision-resolution chain of records and thus close the chain without leaving any deleted records. One such procedure is shown in the aforementioned text by Knuth at page 527. Unfortunately, such non-contaminating procedures, due to the necessity for successive probes into the storage space, take so much time that they can be used only when the data base is off line and hence not available for accessing.

The problem, then, is to provide the speed of access of hashing techniques for large and heavily used information storage systems having expiring data and, at the same time, prevent the large-scale contamination which normally results from expired records in such large and heavily used systems.

Summary of the Invention

In accordance with the illustrative embodiment of the invention, these and other problems are overcome by using a garbage collection procedure "on the fly" while other types of access to the storage space are taking place. In particular, during normal data insertion or retrieval probes into the data store, the expired, obsolete records are identified and removed in the neighborhood of the probe. Specifically, expired or obsolete records in the collision-resolution chain including the record to be accessed are removed as part of the normal retrieval procedure.

This incremental garbage collection technique has the decided advantage of automatically eliminating contamination caused by obsolete or expired records without requiring that the data base be taken off-line for such garbage collection. This is particularly important for data bases requiring rapid access and continuous availability to the user population.

Brief Description of the Drawing

A complete understanding of the present invention may be gained by considering the following detailed description in conjunction with the accompanying drawing, in which:

FIG. 1 shows a general block diagram of a computer system hardware arrangement in which the information storage and retrieval system of the present invention might be implemented;

FIG. 2 shows a general block diagram of a computer system software arrangement in which the information storage and retrieval system of the present invention might find use;

FIG. 3 shows a general flow chart for table searching operation which might be used in a hashed storage system in accordance with the present invention;

FIG. 4 shows a general flow chart for a garbage collecting remove procedure which forms part of the table searching operation of FIG. 3;

FIG. 5 shows a general flow chart for record insertion operations which might be used in a hashed storage system in accordance with the present invention;

FIG. 6 shows a general flow chart for a record retrieval operation for use in a hashed storage system in accordance with the present invention; and

FIG. 7 shows a general flow chart for a record deletion operation which might be used in the hashed storage system in accordance with the present invention.

To facilitate reader understanding, identical reference numerals are used to designate elements common to the figures.

Detailed Description

Referring more particularly to FIG. 1 of the drawings, there is shown a general block diagram of a computer hardware system comprising a Central Processing Unit (CPU) 10 and a Random Access Memory (RAM) unit 11. Computer programs stored in the RAM 11 are accessed by

CPU 10 and executed, one instruction at a time, by CPU 10. Data, stored in other portions of RAM 11, are operated upon by the program instructions accessed by CPU 10 from RAM 11, all in accordance with well-known data processing techniques.

Central Processing Unit (CPU) 10 also controls and accesses a disk controller unit 12 which, in turn, accesses digital data stored on one or more disk storage units such as disk storage unit 13. In normal operation, programs and data are stored on disk storage unit 13 until required by CPU 10. At this time, such programs and data are retrieved from disk storage unit 13 in blocks and stored in RAM 11 for rapid access.

Central Processing Unit (CPU) 10 also controls an Input-Output (IO) controller 14 which, in turn, provides access to a plurality of input devices such as CRT (cathode ray tube) terminal 15, as well as a plurality of output devices such as printer 16. Terminal 15 provides a mechanism for a computer operator to introduce instructions and commands into the computer system of FIG. 1, and may be supplemented with other input devices such as card and tape readers, remotely located terminals, optical readers and other types of input devices. Similarly, printer 16 provides a mechanism for displaying the results of the operation of the computer system of FIG. 1 for the computer user. Printer 16 may similarly be supplemented by line printers, cathode ray tube displays, phototypesetters, graphical plotters and other types of output devices.

The constituents of the computer system of FIG. 1 and their cooperative operation are well-known in the art and are typical of all computer systems, from small personal computers to large main frame systems. The architecture and operation of such systems are well-known and, since they form no part of the present invention, will not be further described here.

In FIG. 2 there is shown a graphical

representation of a typical software architecture for a computer system such as that shown in FIG. 1. The software of FIG. 2 comprises an access mechanism 20 which, for simple personal computers, may comprise no more than turning the system on. In larger systems, providing service to a larger number of users, login and password procedures would typically be implemented in access mechanism 20. Once access mechanism 20 has completed the login procedure, the user is placed in the operating system environment 21. Operating system 21 coordinates the activities of all of the hardware components of the computer system (shown in FIG. 1) and provides a number of utility programs 22 of general use to the computer user. Utilities 22 might, for example, comprise assemblers and compilers, mathematical routines, basic file handling routines and system maintenance facilities.

Many computer software systems also include a data base manager program 23 which controls access to the data records in a data base 24. Data base 24 may, for example, reside on a disk storage unit or units such as disk storage unit 13 of FIG. 1. User application programs such as application program 25 then use the data base manager program 23 to access data base records in data base 24 for adding, deleting and modifying data records. It is the efficient realization of a data base manager such as data base manager program 23 in FIG. 2 to which the present invention is directed.

Before proceeding to a description of one embodiment of the present invention, it is first useful to discuss hashing techniques in general. Hashing techniques have been used classically for very fast access to static, short term data such as a compiler symbol table. Typically, in such storage tables, deletions are infrequent and the need for the storage table disappears quickly.

In some common types of data storage systems, data records become obsolete merely by the passage of time

or by the occurrence of some event. If such expired, lapsed or obsolete records are not removed from the storage table, they will, in time, seriously degrade or contaminate the performance of the retrieval system. Contamination arises because of the ever-increasing need to search longer and longer chains of record locations, many of which are expired, to reach a desired location.

More particularly, a hash table can be described as a logically contiguous, circular list of consecutively numbered, fixed-sized storage units, called cells, each capable of storing a single item called a record. Each record contains a distinguishing field, called the key, which is used as the basis for storing and retrieving the associated record. The keys throughout the hash table data base are distinct and unique for each record. Hashing functions which associate keys with storage addresses are usually not one-to-one in that they map many distinct keys into the same location.

To store a new record, a cell number is generated by invoking the hashing function on the key for the new record. If this cell location is not occupied, the new record is stored there. If this cell location is occupied, a collision has occurred and the new record must be stored elsewhere, in an overflow area, using an appropriate collision-resolution technique. A common collision-resolution strategy, which will be described here, is known as linear probing under open addressing. Open addressing means that the overflow area is the entire hash table itself. Linear probing indicates sequential scanning of cells beginning with the next cell, recalling that the storage table is viewed circularly. The collision is resolved by storing the record in the first unoccupied cell found.

To retrieve a record, the key is hashed to generate a cell location. If the record is not there (the keys do not match), searching continues following the same forward path as record storage. An empty cell terminates

the retrieval procedure, which has then failed to find the record to be retrieved.

In FIG. 3 there is shown a flowchart of a search table procedure for searching the hash table preparatory to inserting, retrieving or deleting a record. The hash table may, for example, comprise the data base 24 of FIG. 2 and the search table procedure of FIG. 3 comprise a portion of the data base manager 23 of FIG. 2. Starting in box 30 of the search table procedure of FIG. 3, the search key of the record being searched for is hashed in box 31 to provide the address of a cell. In box 32, the empty cell just past the end of the search chain of non-empty cells is located, i.e., the first succeeding unoccupied cell is found. In box 33, the procedure moves one cell backward from the current cell position (now at the end of the chain). Decision box 34 examines the cell to determine whether the cell is empty or not. If the cell tested in decision box 34 is empty, decision box 35 is entered to determine if a key match was previously found in decision box 41 (as will be described below). If so, the search is successful and returns success in box 36 and terminates in terminal box 39. If not, box 37 is entered where the location of the empty cell is saved for possible record insertion. In box 38 failure is returned since an empty cell was found before a cell with a matching key. The procedure again terminates in box 39.

If the cell tested in decision box 34 is not empty, decision box 40 is entered to determine if the record in that cell has expired. This is determined by comparing some portion of the contents of the record to some external condition. A timestamp in the record, for example, could be compared with the time-of-day. Alternatively, the occurrence of an event can be compared with a field identifying that event in the record. In any event, if the record has not expired, decision box 41 is entered to determine if the key in this record matches the

search key. If it does, the cell location is saved in box 42 and the procedure returns to box 33. If the record key does not match the search key, the procedure returns directly to box 33.

If decision box 40 determines that the record has expired, box 43 is entered to perform a non-contaminating deletion of the expired record, as will be described in connection with FIG. 4. In general, the procedure of box 43 (FIG. 4) operates to move a record further toward the end of the chain into the position of the record which has expired, thereby removing the expired record and, at the same time, closing the search chain.

It can be seen that the search table procedure of FIG. 3 operates to examine the entire chain of records of which the searched-for record is a part, and to delete expired records by chain-filling rather than by marking such records as deleted. In this way, contamination of the storage space by expired records is removed in the vicinity of each new table search. If contamination becomes too large even with such automatic garbage collection, then the insertion of new records can be inhibited until the search table procedure has had a chance to remove a sufficient number of expired records to render the operation of the system sufficiently efficient.

The search table procedure illustrated generally in FIG. 3 is implemented in the Appendix as PASCAL-like pseudocode. Source code suitable for compilation and execution on any standard hardware and software computing system can readily be devised from this pseudocode and the flowcharts of the figures by any person of ordinary skill in the art.

In FIG. 4 there is shown a flowchart of a remove procedure which removes records from the database, either records to be deleted or expired records. In general, this is accomplished by traversing the chain of the record to be removed in a forward direction searching for a record whose key hashes at or behind the cell to be

removed. When such a record is found, it is copied to the cell of the record to be removed. The copied record is then taken as the record to be removed and the process is continued until the end of the search chain is reached. In box 54, the final copied record is marked empty prior to teminating the procedure. The remove procedure of FIG. 4 might comprise a portion of the data base manager program 23 of FIG. 2.

Starting at starting box 50 of FIG. 4, the procedure is entered with the location of a cell to be removed which is called the base cell. Initially, box 51 is entered where the load count in the table is adjusted to reflect the removal of one record. The load, of course, is the number of occupied cells. As previously noted, the value of this load can be used to disable the insertion of new records until the load has reached a low enough value to permit efficient searching. In box 52, the procedure of FIG. 4 advances to the next cell in the chain beyond the base cell. In decision box 53 this cell is tested to see if it is empty. If it is empty, the end of the chain has been reached and box 54 is entered to mark the base cell as empty. Decision box 55 is then entered to determine if a record was found (by the search table procedure) which matched the search key and, if so, the procedure is terminated in terminal box 56. If a matching record was not found, decision box 57 is entered to determine if the base cell is ahead of the hash location of the search key. If not, the procedure is terminated in box 56. If the base cell does hash ahead of the search record, then the base cell can be used for storing a new record. In box 58, the location of this empty cell is therefore saved as a possible insertion site.

Returning to box 53, if the next cell is not empty, box 59 is entered to determine if the record in this cell hashes ahead of the base cell. If so, box 52 is re-entered to advance to the next cell in the chain. If

this next cell hashes at or behind the base cell, however, box 60 is entered to copy the contents of this next cell to the base cell, thereby obliterating (removing) the base cell contents. Box 61 is then entered to test if the search table procedure found a matching record. If not, box 52 is re-entered to advance to the next cell. If a matching record was found, decision box 62 is entered to test if the matching record is the base cell record. If not, box 52 is re-entered to advance to the next cell. If the matching record is the base cell, however, box 63 is entered to store the location of the former base cell as the position of the matching record and then box 52 is re-entered to advance to the next cell in the search chain.

It can be seen that the procedure of FIG. 4 operates to examine the entire search chain and to move records from later positions in the chain to vacated positions in the chain such that the chain is entirely closed at the end of the procedure. That is, no empty cells are left to erroneously break up a search chain. As noted in connection with FIG. 3, expired records are subjected to the remove procedure of FIG. 4. As will be noted in connection with FIG. 7, records to be deleted from the data base are also subjected to the remove procedure of FIG. 4.

The remove procedure illustrated generally in FIG. 4 is implemented in the Appendix as PASCAL-like pseudocode. Source code suitable for compilation and execution on any standard hardware and software computing system can readily be devised from this pseudocode and the flowchart of FIG. 4 by any person of ordinary skill in the art.

In FIG. 5 there is shown a detailed flowchart of an insert procedure suitable for use in the information storage and retrieval system of the present invention. The insert procedure of FIG. 5 begins as starting box 70 from which box 71 is entered. In box 71, the search table

procedure of FIG. 3 is invoked with the search key of the record to be inserted. As noted in connection with FIG. 3, the search table procedure locates the target cell location and, if part of a search chain, removes all expired cells from that search chain. Decision box 72 is then entered where it is determined whether or not the search table procedure found a record with a matching key. If so, box 73 is entered where the record to be inserted in put into the storage table in the position of the old record with a matching key. In box 74, the insert procedure reports that the old record has been replaced by the new record and the procedure is terminated in terminal box 75.

Returning to decision box 72, if a matching record is not found, decision box 76 is entered to determine if the table load is below a preselected threshold (typically about 75% of the table capacity). If the load is not below the threshold, the storage table is too full to be access efficiently, and box 77 is entered to report that the the table is full and the record cannot be inserted. The procedure then terminates in terminal box 75. If the load is below the threshold, box 78 is entered where the record to be inserted is placed in the empty cell position found by the search table procedure. In box 79, the load is adjusted to reflect the addition of one record to the storage table, the procedure reports that the record was inserted in box 80 and the procedure terminated in box 75.

The insert procedure illustrated generally in FIG. 5 is implemented in the Appendix as PASCAL-like pseudocode. Source code suitable for compilation and execution on any standard hardware and software computing system can readily be devised from this pseudocode and the flowcharts of the FIG. 5 by any person of ordinary skill in the art.

In FIG. 6 there is show a detailed flowchart of a retrieve procedure which is used to retrieve a record

from the data base 24 of FIG. 2. Starting in box 90, the search table procedure is invoked in box 91, using the key of the record to be retrieved as the search key. In box 92 it is determined if a record with a matching key was found by the search table procedure. If not, box 93 is entered to report failure of the retrieve procedure and the procedure is terminated in box 96. If a matching record was found, box 94 is entered to copy the matching record into a buffer store for processing by the calling program, box 95 is entered to return an indication of successful retrieval and the procedure terminated in box 96.

The pseudo-code for the retrieve procedure of FIG. 6 is included in the Appendix. Executable code for all common hardware and system software arrangements can readily be devised by those skilled in the art from the flowchart and the pseudo-code.

In FIG. 7 there is shown a detailed flowchart of a delete procedure useful for actively removing records from the data base 24 of FIG. 2. Starting at box 100, the procedure of FIG. 7 first invokes the search table procedure of FIG. 3 in box 101, using the key of the record to be deleted as the search key. In box 102, it is determined if the search table procedure was able to locate a record with a matching key. If not, box 103 is entered to report failure of the deletion procedure and the procedure is terminated in box 106. If a matching record was found, as determined by box 102, the remove procedure of FIG. 4 is invoked in box 104. As noted in connection with FIG. 4, this procedure removes the record to be deleted and, at the same time, closes the search chain. Box 105 is then entered to report successful deletion to the calling program and the procedure is terminated in box 106.

The delete procedure illustrated generally in FIG. 7 is implemented in the Appendix as PASCAL-like pseudocode. Source code suitable for compilation and

execution on any standard hardware and software computing system can readily be devised from this pseudocode and the flowchart of FIG. 7 by any person of ordinary skill in the art.

The attached Appendix contains pseudocode listings for all of the programmed functions necessary to implement a data base manager 23 (FIG. 2) operating in accordance with the present invention. These listings follow the flowcharts of FIGS. 3-7 and further explain and elucidate the flowcharts. Any person of ordinary skill in the art will have no difficulty implementing these functions in any desired program language to run on any desired computer hardware configuration.

It should also be clear to those skilled in the art that further embodiments of the present invention may be made by those skilled in the art without departing from the teachings of the present invention.

APPENDIX

Functions Provided

The following functions are made available to the application program:

insert (record: record type)

Returns replaced if a record associated with record.key was found in the table and subsequently replaced.

Returns inserted if a record associated with record.key was not found in the table and the passed record was subsequently inserted.

Returns full if a record associated with record.key was not found in the table and passed record could not be inserted because load factor has reached max load factor.

retrieve (record: record type)

Returns success if record associated with record.key was found in the table and assigned to record.

Returns failure if search was unsuccessful.

delete (record key: record key type)

Returns success if record associated with record key was found in the table and subsequently deleted.

Returns failure if none found.

Definitions

The following formal definitions are required for specifying the insertion, retrieval, and deletion algorithms:

```
const table size            /* size of hash table */

const max load factor       /* 0 ≤ max load factor < 1 */

var table: array[0 .. table size-1] of record type;
                    /* hash table */

var load: 0 .. table size-1;
                    /* number of occupied entries of
                       hash table array (initially 0) */
```

Algorithms

Algorithms for the functions described above are given below:

```
function insert (record: record type):
         (replaced, inserted, full );

  var position: 0 .. table size-1;
                /* position in table to update or
                   insert (returned by search table) */

  begin

    if search table (record.key, position)

       then begin

         table[position] := record;
         return (replaced)

       end

       else if load/table size < max load factor

         then begin

           load := load+1;
           table[position] := record;
           return (inserted)

         end

         else return (full)

  end                /* insert */
```

```
function retrieve (var record: record type):
        (success, failure);

  var position: 0 .. table size-1;
            /* position in table where record
               resides (returned by search table) */

  begin

    if search table (record.key, position)

       then begin

         record := table[position];
         return (success)

       end

       else return (failure)

   end  /* retrieve */
```

```
function delete (record key: record key type):
      (success, failure);

  var position: 0 .. table size-1;
            /* position in table where record
               resides (returned by search table) */

    dummy variable: 0 .. table size-1;
            /* last two arguments to remove are
               not relevant here                  */


  begin

    if search table (record key, position)

      then begin

        remove (position, true, dummy variable,
              dummy variable);

        return (success)

      end

      else return (failure)

  end    /* delete */
```

```
function search table (record key: record key type;
        var position: 0 .. table size-1): boolean;

  /* search table for record key and delete expired
     expired records in target chain; position is set to
     index of found record or appropriate empty cell */

  var i: 0 .. table size-1;
       /* used for scanning chain,
          both forwards & backwards */

      pos empty: 0 .. table size-1;
       /* index of leftmost empty cell
          to right of position */

      is rec found: boolean;
       /* indicates whether search is successful */


  begin

    position := hash (record key);
    is rec found :=  false;

    if table[position] is not empty then

        begin
          i := position; /* loop initialization */
          repeat         /* scan forward to end of chain
                            containing table[position]  */

              i := (i+1) mod table size

          until (table[i] is empty);

          pos empty := i;
```

```
            i := (i-1+table size) mod table size;

            while (table[i] is not empty) do
                  /*scan chain in reverse,
                    deleting expired entries */

5               begin

                  if table[i] is expired
                            then remove (i, is rec found,
                                 position, pos empty)

10                          else if table[i].key = record key

                                    then begin

                                       is rec found := true;
15                                     position := i

                                    end;

                  i := (i-1+table size) mod table size

                end;   /* while  */

20            if not is rec found then position := pos empty

              end;    /* then  */

          return (is rec found)

      end                    /* search table  */
```

```
procedure remove (cell to del:
 0 .. table size-1; is rec found: boolean;
 var pos of search rec, pos empty: 0 .. table size-1;

          /* Delete table[cell to del] */

 5   var i, j: 0 .. table size-1;

     begin

        load := load-1;

        do forever

10        i := cell to del;
                    /* save position of emptied slot */

          repeat          /* scan forward looking for a
                             record to fill hole in chain */

             cell to del := (cell to del+1) mod table size;

15           if table[cell to del] is empty

                then begin

                   table[i] := empty;

                   if not is rec found then

20                    if (pos of search rec ≤ i < pos empty)
                      or (i< pos empty < pos of search rec)
                      or (pos empty< pos of search rec ≤
                      i) then pos empty := i;

                   return
```

```
                end;

                j := hash (table[cell to del].key)

            until (j ≤ i < cell to del)
            or (i < cell to del < j)
            or (cell to del< j ≤ i);

            table[i] := table[cell to del];
            /* use table[cell to del] to plug hole in chain */

            if (is rec found) and
            (pos of search rec = cell to del)
            then pos of search rec := i

        end

    end                    /* remove */
```

What is claimed is:

1. An information storage and retrieval system using hashing techniques to provide rapid access to the records of said system and utilizing a linear probing technique to store records with the same hash address, said system comprising

a record search means utilizing a search key to access a chain of records having the same hash address,

means for removing all expired records from said chain of records, and

means, utilizing said record search means, for inserting, retrieving and deleting records from said system.

2. The information storage and retrieval system according to claim 1 further comprising

means for recursively moving a record from a later position in said chain of records into the position of one of said expired records.

3. The information storage and retrieval system according to claim 1 further including

means for inhibiting the insertion of new records into said system when the available storage space falls below a preselected value.

4. The information storage and retrieval system according to claim 3 wherein

means for rehabling the insertion of new records into said system when the available storage space rises above said preselected value.

5. An automatically decontaminating hashed storage table comprising

means for accessing said storage table for inserting, retrieving and deleting records, and

means for automatically removing expired records from said table each time said table is accessed.

6. A method for storing and retrieving information records using hashing techniques to provide rapid access to said records and utilizing a linear probing technique to store records with the same hash address, said system comprising

accessing a chain of records having the same hash address,

removing all expired records from said chain of records, and

utilizing said record search means, for inserting, retrieving and deleting records from said system.

7. The method according to claim 6 further comprising the step of

moving a record from a later position in said chain of records into the position of one of said expired records.

8. The method according to claim 6 further including the step of

inhibiting the insertion of new records into said system when the available storage space falls below a preselected value.

9. The method according to claim 8 further comprising the step of

re-enabling the insertion of new records into said system when the available storage space rises above said preselected value.

10. A method for automatically decontaminating a hashed storage table comprising the steps of

accessing said storage table for inserting, retrieving and deleting records, and

automatically removing expired records from said table each time said table is accessed.

Abstract of the Disclosure

A method and apparatus for performing storage and retrieval in an information storage system is disclosed which uses the hashing technique. In order to prevent contamination of the storage medium by automatically expiring records, a garbage collection technique is used which removes all expired records in the neighborhood of a probe into the data storage system. More particularly, each probe for insertion, retrieval or deletion of a record is an occasion to search the entire chain of records found for expired records and then removing them and closing the chain. This garbage collection automatically removes expired record contamination in the vicinity of the probe, thereby automatically decontaminating the storage space. Because no long term contamination can build up in the present system, it is useful for large data bases which are heavily used and which require the fast access provided by hashing.

FIG. 1

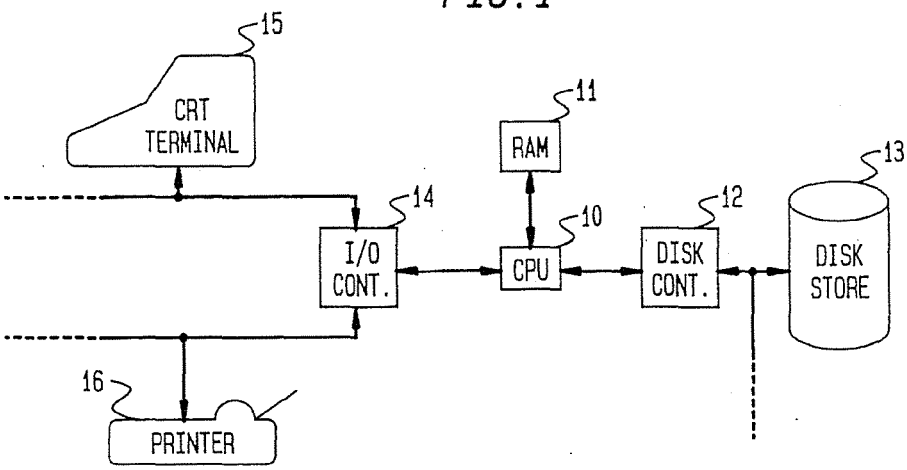

15
CRT
TERMINAL
11
RAM
13
14
10
12
I/O
CONT.
CPU
DISK
CONT.
DISK
STORE
16
PRINTER

FIG. 2

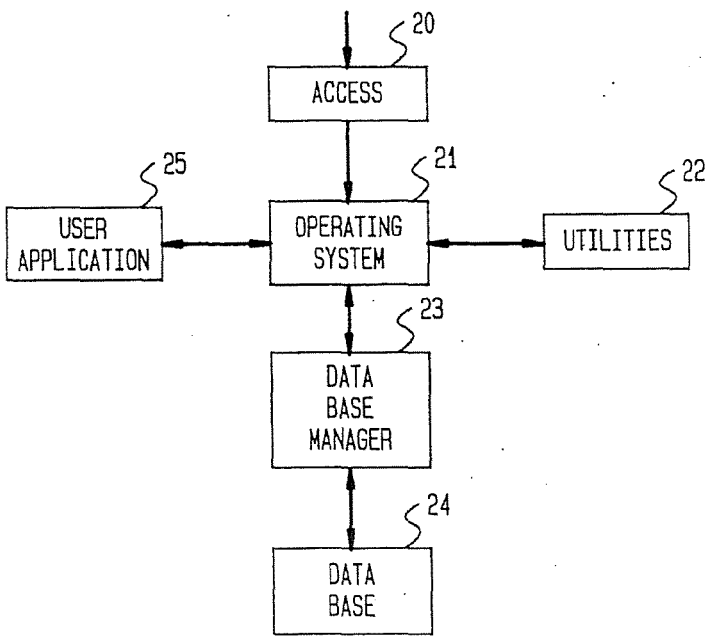

20
ACCESS
25
21
22
USER
APPLICATION
OPERATING
SYSTEM
UTILITIES
23
DATA
BASE
MANAGER
24
DATA
BASE

FIG. 3

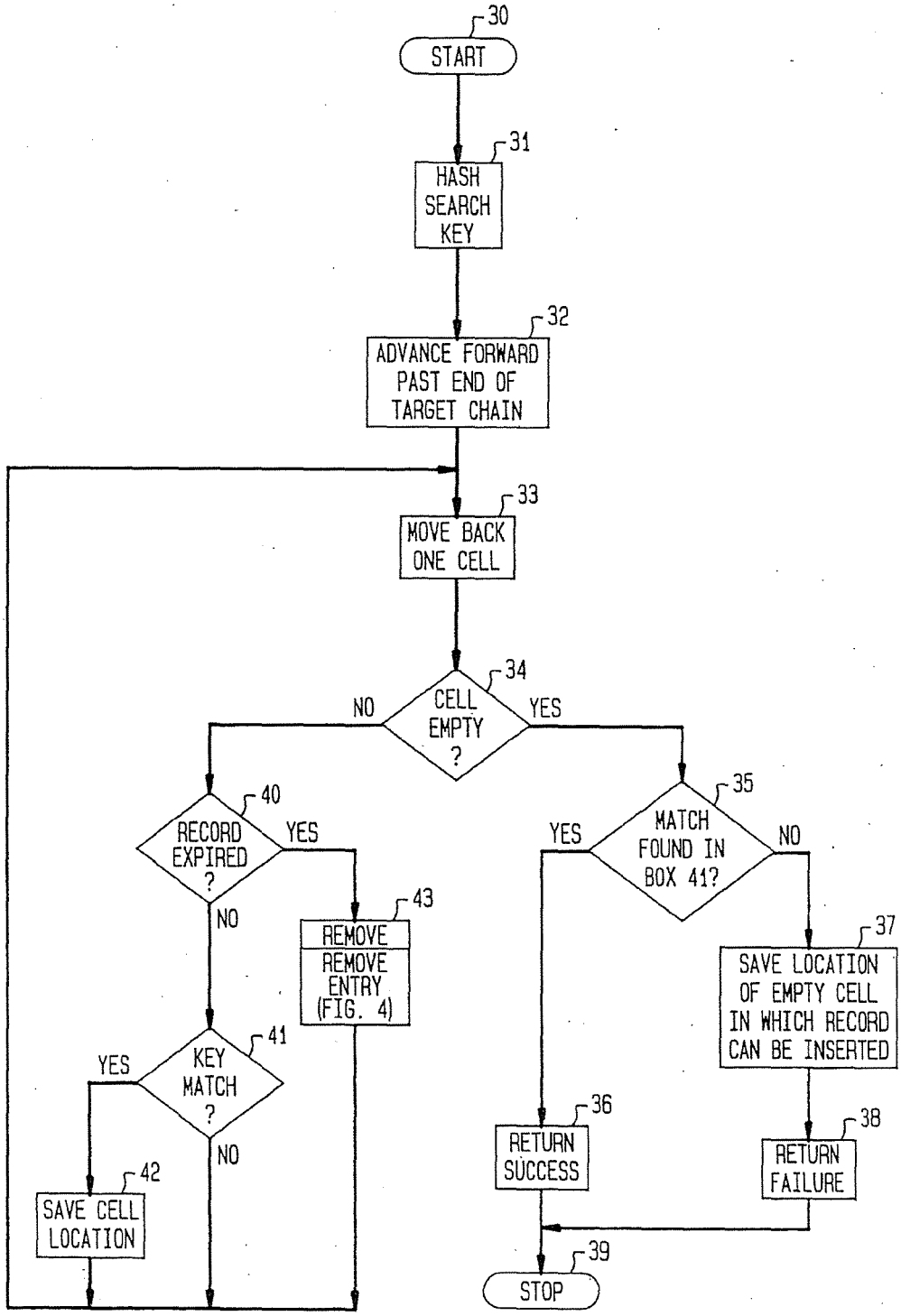

30
START
31
HASH SEARCH KEY
32
ADVANCE FORWARD PAST END OF TARGET CHAIN
33
MOVE BACK ONE CELL
34
CELL EMPTY ?
NO
YES
40
RECORD EXPIRED ?
YES
NO
43
REMOVE
REMOVE ENTRY (FIG. 4)
41
KEY MATCH ?
YES
NO
42
SAVE CELL LOCATION
35
MATCH FOUND IN BOX 41?
YES
NO
36
RETURN SUCCESS
37
SAVE LOCATION OF EMPTY CELL IN WHICH RECORD CAN BE INSERTED
38
RETURN FAILURE
39
STOP

FIG. 4

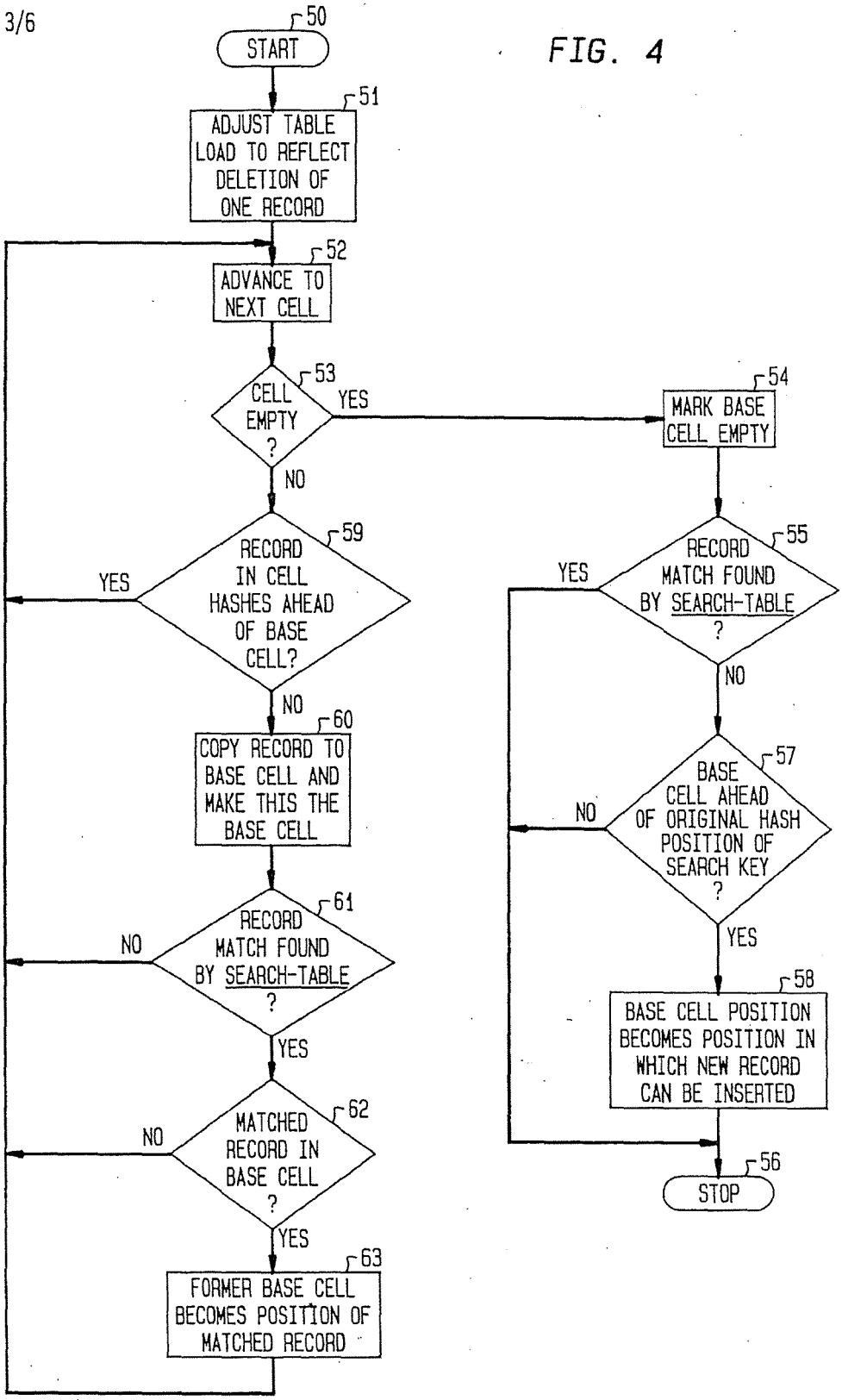

50
START
51
ADJUST TABLE LOAD TO REFLECT DELETION OF ONE RECORD
52
ADVANCE TO NEXT CELL
53
CELL EMPTY ?
YES
54
MARK BASE CELL EMPTY
NO
59
RECORD IN CELL HASHES AHEAD OF BASE CELL?
YES
NO
60
COPY RECORD TO BASE CELL AND MAKE THIS THE BASE CELL
61
RECORD MATCH FOUND BY SEARCH-TABLE ?
NO
YES
62
MATCHED RECORD IN BASE CELL ?
NO
YES
63
FORMER BASE CELL BECOMES POSITION OF MATCHED RECORD
55
RECORD MATCH FOUND BY SEARCH-TABLE ?
YES
NO
57
BASE CELL AHEAD OF ORIGINAL HASH POSITION OF SEARCH KEY ?
NO
YES
58
BASE CELL POSITION BECOMES POSITION IN WHICH NEW RECORD CAN BE INSERTED
56
STOP

FIG. 5

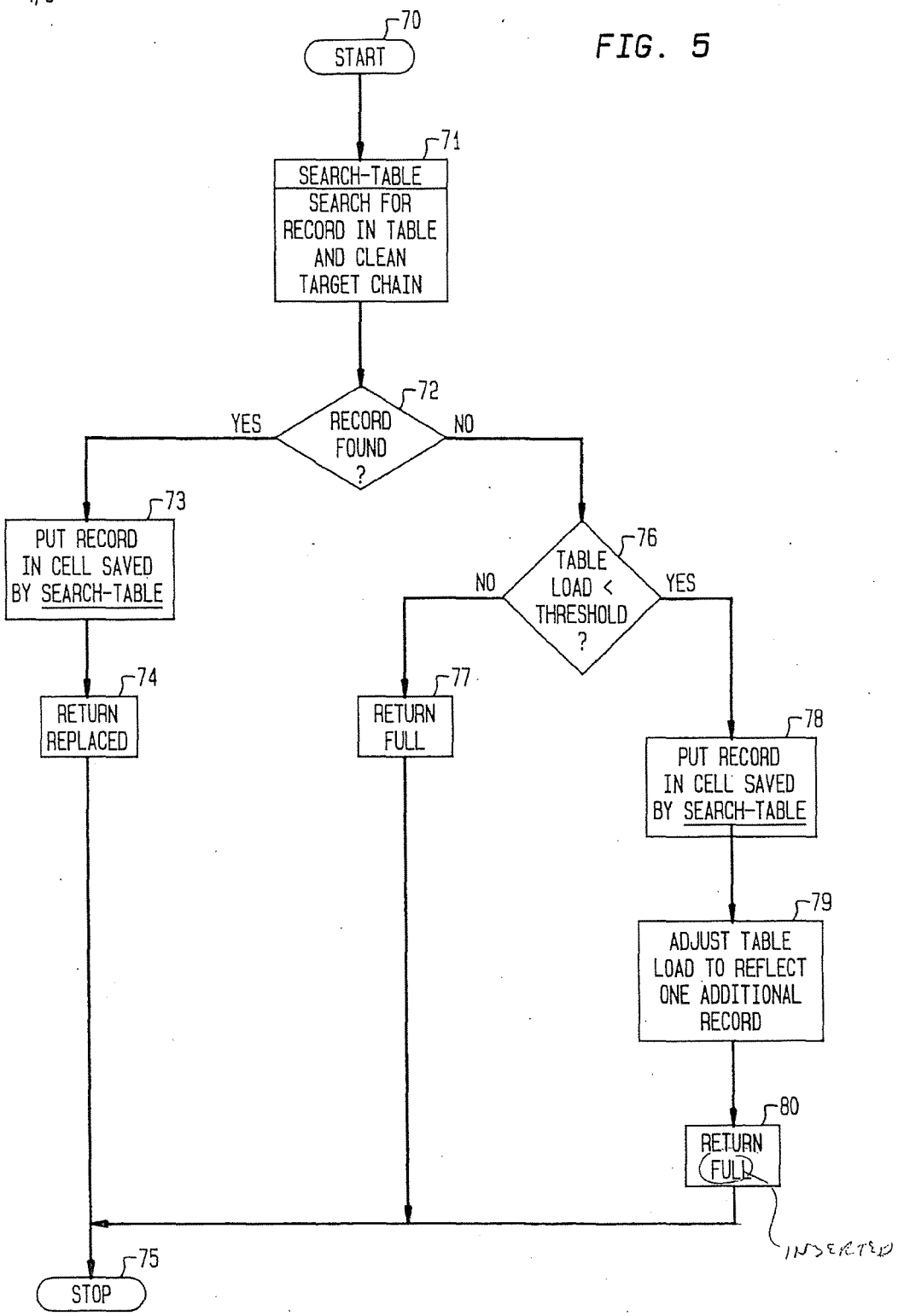

70
START
71
SEARCH-TABLE
SEARCH FOR RECORD IN TABLE AND CLEAN TARGET CHAIN
72
RECORD FOUND ?
YES
NO
73
PUT RECORD IN CELL SAVED BY SEARCH-TABLE
74
RETURN REPLACED
76
TABLE LOAD < THRESHOLD ?
NO
YES
77
RETURN FULL
78
PUT RECORD IN CELL SAVED BY SEARCH-TABLE
79
ADJUST TABLE LOAD TO REFLECT ONE ADDITIONAL RECORD
80
RETURN FULL
INSERTED
75
STOP

FIG. 6

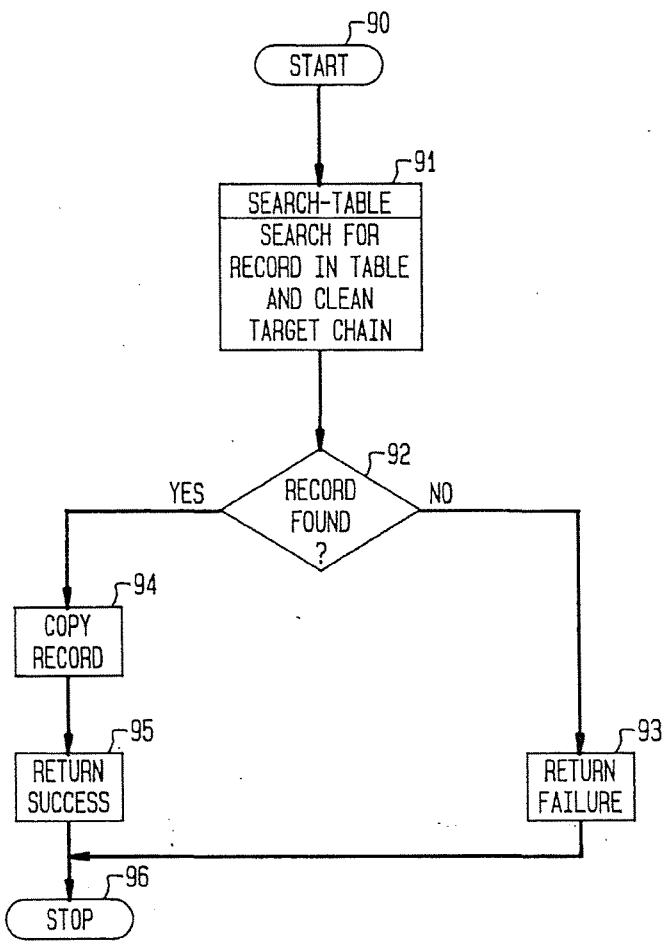
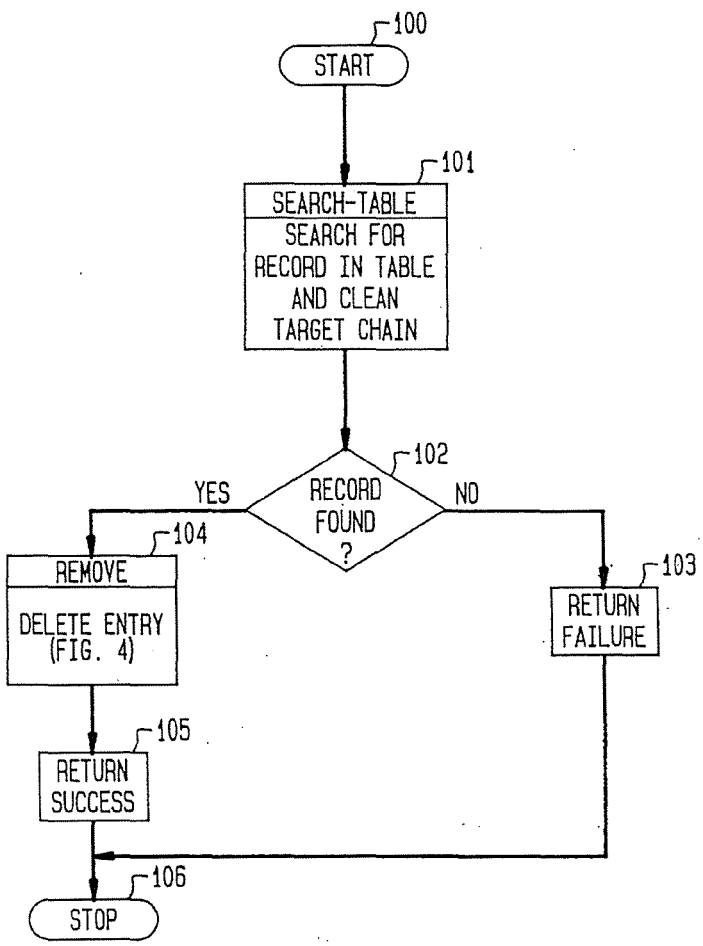

90
START
91
SEARCH-TABLE
SEARCH FOR RECORD IN TABLE AND CLEAN TARGET CHAIN
92
RECORD FOUND ?
YES
NO
94
COPY RECORD
95
RETURN SUCCESS
93
RETURN FAILURE
96
STOP

FIG. 7


100
START
101
SEARCH-TABLE
SEARCH FOR RECORD IN TABLE AND CLEAN TARGET CHAIN
102
RECORD FOUND ?
YES
NO
104
REMOVE
DELETE ENTRY (FIG. 4)
103
RETURN FAILURE
105
RETURN SUCCESS
106
STOP

APPLICATION APPROVAL FOR FILING

Proposed Class 364 Subclass 200	Date to Inventor 1-7-87

Inventor-Case No. (Alphabetical Listing - Patent and Trademark Office Requirement)

R. M. NEMES CASE 2

Title	Equations/Tables Approved: -
METHODS AND APPARATUS FOR INFORMATION STORAGE AND RETRIEVAL	RM, 1/8/88 Attorney

Kind of Application	Assignee	No. of Claims	No. of Dwg.Sheets
☒ New ☐ Continuation ☐ Design ☐ Reissue ☐ Divisional	☒ Bellcore ☐	4 Independent 6 Dependent	6 / No. of Figs. 7

Approved For Filing (Inventor(s) and Attorneys to Sign and Date)

By	Richard Michael Nemes		
Date	January 7, 1988		
By	K.O. Smith		
Date	1/7/88		
By			
Date			

NOTE TO INVENTOR → Under Inventor's Verification: Please approve each correct item or note change, if any. If not available for signature during next two weeks, note date of availability in address portion. If planning to move, note "moving" and approximate date.

Intellectual Property Matters Records	Inventor's Verification
Inventor RICHARD MICHAEL NEMES	Inventor OK
Vocation COMPUTER SCIENTIST	Vocation OK
Citizenship UNITED STATES OF AMERICA	Citizenship OK
Residence Address KINGS COUNTY BROOKLYN, NEW YORK	Residence Address OK
Post Office Address (include zip code) 1730 EAST 14TH ST, APT. 6G BROOKLYN, NEW YORK 11229	Post Office Address OK
Inventor	Inventor
Vocation	Vocation
Citizenship	Citizenship
Residence Address	Residence Address
Post Office Address (include zip code)	Post Office Address

Pt. 7 2-85

(over)

DRAFT

Methods and Apparatus for Information Storage and Retrieval

Technical Field

This invention relates to information storage and retrieval systems and, more particularly, to the use of hashing techniques in such systems.

Background of the Invention

Information or data stored in a computer-controlled storage mechanism can be retrieved by searching for a particular key in the stored records. The Records with its ~~a stored~~ key matching the search key ~~are~~ is then retrieved. Such searching techniques require repeated accesses or probes into the storage mechanism to perform the key comparisons. In large storage and retrieval systems, such searching, even if augmented by efficient search algorithms such as a binary search, often requires an excessive amount of time.

Another well-known and much faster method for storing and retrieving information from computer store involves the use of so-called "hashing" techniques. These techniques are also sometimes called scatter-storage or key-transformation techniques. In a system using hashing, the key is operated upon (by a hashing function) to produce a storage address in the storage space (called the hash table). This storage address is then used to access the desired storage location directly with fewer storage accesses or probes than sequential or binary searches. Hashing techniques are described in the classic text by D. Knuth entitled The Art of Computer Programming, Volume 3, Sorting and Searching, pp.506-549, Addison-Wesley, Reading, Massachusetts, 1973.

Hashing functions are designed to translate the universe of keys into addresses uniformly distributed throughout the hash table. Typical hashing operations include truncation, folding, transposition and modulo arithmetic. A disadvantage of hashing techniques is that more than one key can translate into the same storage

address, causing "collisions" in storage or retrieval operations. Some form of collision-resolution strategy (sometimes called "rehashing") must therefore be provided. For example, the simple strategy of searching forward from the initial storage address to the first empty storage location will resolve the collision. This latter technique is called linear probing. If the hash table is considered to be circular so that addresses beyond the end of the table map back to the beginning of the table, then the linear probing is done with "open addressing," i.e., with the entire hash table as overflow space in the event that a collision occurs.

Some forms of data records have a limited lifetime after which they become obsolete. Scheduling activities, for example, involves records which become obsolete after the scheduled activity has occurred. Such record storage locations cannot be simply emptied since this location may be a link in a chain of locations previously created during a collision-resolution procedure. The classic solution to this problem is to mark the record as "deleted" rather than as "empty," and to leave the record in place. In time, however, the storage space can become contaminated by an excessive number of deleted or obsolete storage locations that must be searched to locate desired records. With the passage of time, such storage contamination can reduce the performance of retrieval operations below acceptable levels. Problems of this type are discussed in considerable detail in Data Structures and Program Design, by R. L. Kruse, Prentice-Hall, Englewood Cliffs, New Jersey, 1984, pp. 112-126, and Data Structures with Abstract Data Types and PASCAL, by D. F. Stubbs and N. W. Webre, Brooks/Cole Publishing, Monterey, California, 1985, pp. 310-336.

In the prior art, such storage space contamination was avoided by deletion procedures that eliminated deleted records by replacing the deleted record

with another record in the collision-resolution chain of records and thus close the chain without leaving any deleted records. One such procedure is shown in the aforementioned text by Knuth at page 527. Unfortunately, such non-contaminating procedures, due to the necessity for successive probes into the storage space, take so much time that they can be used only when the data base is off line and hence not available for accessing.

The problem, then, is to provide the speed of access of hashing techniques for large and heavily used information storage systems having expiring data and, at the same time, prevent the large-scale contamination which normally results from expired records in such large and heavily used systems.

Summary of the Invention

In accordance with the illustrative embodiment of the invention, these and other problems are overcome by using a garbage collection procedure "on the fly" while other types of access to the storage space are taking place. In particular, during normal data insertion or retrieval probes into the data store, the expired, obsolete records are identified and removed in the neighborhood of the probe. Specifically, expired or obsolete records in the collision-resolution chain including the record to be accessed are removed as part of the normal retrieval procedure.

This incremental garbage collection technique has the decided advantage of automatically eliminating contamination caused by obsolete or expired records without requiring that the data base be taken off-line for such garbage collection. This is particularly important for data bases requiring rapid access and continuous availability to the user population.

Brief Description of the Drawing

A complete understanding of the present invention may be gained by considering the following detailed description in conjunction with the accompanying

drawing, in which:

FIG. 1 shows a general block diagram of a computer system hardware arrangement in which the information storage and retrieval system of the present invention might be implemented;

FIG. 2 shows a general block diagram of a computer system software arrangement in which the information storage and retrieval system of the present invention might find use;

FIG. 3 shows a general flow chart for table searching operation which might be used in a hashed storage system in accordance with the present invention;

FIG. 4 shows a general flow chart for a garbage collecting remove procedure which forms part of the table searching operation of FIG. 3;

FIG. 5 shows a general flow chart for record insertion operations which might be used in a hashed storage system in accordance with the present invention;

FIG. 6 shows a general flow chart for a record retrieval operation for use in a hashed storage system in accordance with the present invention; and

FIG. 7 shows a general flow chart for a record deletion operation which might be used in the hashed storage system in accordance with the present invention.

To facilitate reader understanding, identical reference numerals are used to designate elements common to the figures.

Detailed Description

Referring more particularly to FIG. 1 of the drawings, there is shown a general block diagram of a computer hardware system comprising a Central Processing Unit (CPU) 10 and a Random Access Memory (RAM) unit 11. Computer programs stored in the RAM 11 are accessed by CPU 10 and executed, one instruction at a time, by CPU 10. Data, stored in other portions of RAM 11, are operated upon by the program instructions accessed by CPU 10 from

RAM 11, all in accordance with well-known data processing techniques.

Central Processing Unit (CPU) 10 also controls and accesses a disk controller unit 12 which, in turn, accesses digital data stored on one or more disk storage units such as disk storage unit 13. In normal operation, programs and data are stored on disk storage unit 13 until required by CPU 10. At this time, such programs and data are retrieved from disk storage unit 13 in blocks and stored in RAM 11 for rapid access.

Central Processing Unit (CPU) 10 also controls an Input-Output (IO) controller 14 which, in turn, provides access to a plurality of input devices such as CRT (cathode ray tube) terminal 15, as well as a plurality of output devices such as printer 16. Terminal 15 provides a mechanism for a computer operator to introduce instructions and commands into the computer system of FIG. 1, and may be supplemented with other input devices such as card and tape readers, remotely located terminals, optical readers and other types of input devices. Similarly, printer 16 provides a mechanism for displaying the results of the operation of the computer system of FIG. 1 for the computer user. Printer 16 may similarly be supplemented by line printers, cathode ray tube displays, phototypesetters, graphical plotters and other types of output devices.

The constituents of the computer system of FIG. 1 and their cooperative operation are well-known in the art and are typical of all computer systems, from small personal computers to large main frame systems. The architecture and operation of such systems are well-known and, since they form no part of the present invention, will not be further described here.

In FIG. 2 there is shown a graphical representation of a typical software architecture for a computer system such as that shown in FIG. 1. The software of FIG. 2 comprises an access mechanism 20 which,

for simple personal computers, may comprise no more than turning the system on. In larger systems, providing service to a larger number of users, login and password procedures would typically be implemented in access mechanism 20. Once access mechanism 20 has completed the login procedure, the user is placed in the operating system environment 21. Operating system 21 coordinates the activities of all of the hardware components of the computer system (shown in FIG. 1) and provides a number of utility programs 22 of general use to the computer user. Utilities 22 might, for example, comprise assemblers and compilers, mathematical routines, basic file handling routines and system maintenance facilities.

Many computer software systems also include a data base manager program 23 which controls access to the data records in a data base 24. Data base 24 may, for example, reside on a disk storage unit or units such as disk storage unit 13 of FIG. 1. User application programs such as application program 25 then use the data base manager program 23 to access data base records in data base 24 for adding, deleting and modifying data records. It is the efficient realization of a data base manager such as data base manager program 23 in FIG. 2 to which the present invention is directed.

Before proceeding to a description of one embodiment of the present invention, it is first useful to discuss hashing techniques in general. Hashing techniques have been used classically for very fast access to static, short term data such as a compiler symbol table. Typically, in such storage tables, deletions are infrequent and the need for the storage table disappears quickly. ~~A badly hashed table therefore has only a very short lifetime.~~

~~If it is desired to take advantage of the fast access provided by hashing in long term dynamic data, problems arise.~~ In some common types of data storage systems, data records become obsolete merely by the

passage of time or by the occurrence of some event. If such expired, lapsed or obsolete records are not removed from the storage table, they will, in time, seriously degrade or contaminate the performance of the retrieval system. Contamination arises because of the ever-increasing need to search longer and longer chains of record locations, many of which are expired, to reach a desired location.

More particularly, a hash table can be described as a logically contiguous, circular list of consecutively numbered, fixed-sized storage units, called cells, each capable of storing a single item called a record. Each record contains a distinguishing field, called the key, which is used as the basis for storing and retrieving the associated record. The keys throughout the hash table data base are distinct and unique for each record. Hashing functions which associate keys with storage addresses are usually not one-to-one in that they map many distinct keys into the same location.

To store a new record, a cell number is generated by invoking the hashing function on the key for the new record. If this cell location is not occupied, the new record is stored there. If this cell location is occupied, a collision has occurred and the new record must be stored elsewhere, in an overflow area, using an appropriate collision-resolution technique. A common collision-resolution strategy, which will be described here ~~for convenience~~, is known as linear probing under open addressing. Open addressing means that the overflow area is the entire hash table itself. Linear probing indicates sequential scanning of cells beginning with the next cell, recalling that the storage table is viewed circularly. The collision is resolved by storing the record in the first unoccupied cell found.

To retrieve a record, the key is hashed to generate a cell location. If the record is not there (the keys do not match), searching continues following the same

forward path as record storage. An empty cell terminates the retrieval procedure, which has then failed to find the record to be retrieved.

It is to be understood that the present invention will be described in connection with linear probing with open addressing only for convenience and because such a collision-resolution strategy is very commonly used. The techniques of the present invention can just as readily applied to such other forms of collision-resolution strategies by modifications readily apparent to those skilled in the art.

In FIG. 3 there is shown a flow chart of a search table procedure for searching the hash table preparatory to inserting, retrieving or deleting a record. The hash table may, for example, comprise the data base 24 of FIG. 2 and the search table procedure of FIG. 3 comprise a portion of the data base manager 23 of FIG. 2 Starting in box 30 of the search table procedure of FIG. 3, the search key of the record being searched for is hashed in box 31 to provide the address of a cell. In box 32, the empty cell just past the end of the search chain of non-empty cells is located, i.e., the first succeeding unoccupied cell is found. In box 33, the procedure moves one cell backward from the current cell position (now at the end of the chain). Decision box 34 examines the cell to determine if the cell is empty or not. If the cell tested in decision box 34 is empty, the decision box 35 is entered to determine if ~~the key of that cell matches the search key.~~ a match was previously found in box 41 (to be described below). If so, the search is successful and returns success in box 36 and terminates in terminal box 39. If ~~the key of in the cell tested in box 35 does not match the search key,~~ not box 37 is entered where the location of the empty cell is saved for possible record insertion. In box 38 failure is returned since an empty cell was found before a cell with a matching key. The procedure again terminates in box 39.

If the cell tested in decision box 34 is not

empty, decision box 40 is entered to determine if the record in that cell has expired. This is determined by comparing some portion of the contents of the record to some external condition. A timestamp in the record, for example, could be compared with the time-of-day. Alternatively, the occurrence of an event can be compared with a field identifying that event in the record. In any event, if the record has not expired, decision box 41 is entered to determine if the key in this record matches the search key. If it does, the cell location is saved in box 42 and the procedure returns to box 33. If the record key does not match the search key, the procedure returns directly to box 33.

If decision box 40 determines that the record has expired, box 43 is entered to perform a non-contaminating deletion of the expired record, as will be described in connection with FIG. 4. In general, the procedure of box 43 (FIG. 4) operates to move a record ~~at~~ found the end of the chain into the position of the record which has expired, thereby removing the expired record and, at the same time, closing the search chain.

It can be seen that the search table procedure of FIG. 3 operates to examine the entire chain of records of which the searched-for record is a part, and to delete expired records by chain-filling rather than by marking such records as deleted. In this way, contamination of the storage space by expired records is removed in the vicinity of each new table search. If contamination becomes too large even with such automatic garbage collection, then the insertion of new records can be inhibited until the search table procedure has had a chance to remove a sufficient number of expired records to render the operation of the system sufficiently efficient.

The search table procedure illustrated generally in FIG. 3 is implemented in the Appendix as PASCAL-like pseudocode. Source code suitable for compilation and execution on any standard hardware and software computing

system can readily be devised from this pseudocode and the flowcharts of the figures by any person of ordinary skill in the art.

In FIG. 4 there is shown a flowchart of a remove procedure which removes records from the database, either records to be deleted or expired records. In general, this is accomplished by traversing the chain of the record to be removed in a forward direction searching for a record whose key hashes at or behind the cell to be removed. When such a record is found, it is copied to the cell of the record to be removed. The copied record is then taken as the record to be removed and the process continued until the end of the search chain is reached. The remove procedure of FIG. 4 might comprise a portion of the data base manager program 23 of FIG. 2.

Starting at starting box 50 of FIG. 4, the procedure is entered with the location of a cell to be removed which is called the base cell. Initially, box 51 is entered where the load ~~factor~~ of the table is adjusted to reflect the removal of one record. The load ~~factor~~, of course, is the ~~fractional portion of the total table~~ number of cells which are ~~is~~ occupied with records. As previously noted, this load ~~factor~~ can be used to disable the insertion of new records until the load ~~factor~~ has reached a low enough value to permit efficient searching. In box 52, the procedure of FIG. 4 advances to the next cell in the chain beyond the base cell. In decision box 53 this cell is tested to see if it is empty. If it is empty, the end of the chain has been reached and box 54 is entered to mark the base cell as empty. Decision box 55 is then entered to determine if a record was found (by search table) which matched the search key and, if so, the procedure is terminated in terminal box 56. If a matching record was not found, decision box 57 is entered to determine if the base cell is ahead of the hash location of the search key. If not, the procedure is terminated in box 56. If the base cell does hash ahead of the search record, then the base cell can be used for

storing a new record. In box 58, the location of this empty cell is therefore saved as a possible insertion site.

Returning to box 53, if the next cell is not empty, box 59 is entered to determine if the record in this cell hashes ahead of the base cell. If so, box 52 is re-entered to advance to the next cell in the chain. If this next cell hashes at or behind the base cell, however, box 60 is entered to copy the contents of this next cell to the base cell, thereby obliterating (removing) the base cell contents. Box 61 is then entered to test if the search table procedure found a matching record. If not, box 52 is re-entered to advance to the next cell. If a matching record was found, decision box 62 is entered to test if the matching record is the base cell record. If not, box 52 is re-entered to advance to the next cell. If the matching record is the base cell, however, box 63 is entered to store the location of the former base cell as the position of the matching record and then box 52 is re-entered to advance to the next cell in the search chain.

It can be seen that the procedure of FIG. 4 operates to examine the entire search chain and to move records from later positions in the chain to vacated positions in the chain such that the chain is entirely closed at the end of the procedure. That is, no empty cells are left to erroneously break up a search chain. As noted in connection with FIG. 3, expired records are subjected to the remove procedure of FIG. 4. As will be noted in connection with FIG. 7, records to be deleted from the data base are also subjected to the remove procedure of FIG. 4.

The remove procedure illustrated generally in FIG. 4 is implemented in the Appendix as PASCAL-like pseudocode. Source code suitable for compilation and execution on any standard hardware and software computing system can readily be devised from this pseudocode and the flowchart of FIG. 4 by any person of ordinary skill in the art.

In FIG. 5 there is shown a detailed flowchart of an insert procedure suitable for use in the information storage and retrieval system of the present invention. The insert procedure of FIG. 5 begins as starting box 70 from which box 71 is entered. In box 71, the search table procedure of FIG. 3 is invoked with the search key of the record to be inserted. As noted in connection with FIG. 3, the search table procedure locates the target cell location and, if part of a search chain, removes all expired cells from that search chain. Decision box 72 is then entered where it is determined whether or not the search table procedure found a record with a matching key. If so, box 73 is entered where the record to be inserted is ~~in~~ put into the storage table in the position of the old record with a matching key. In box 74, the insert procedure reports that the old record has been replaced by the new record and the procedure is terminated in terminal box 75.

Returning to decision box 72, if a matching record is not found, decision box 76 is entered to determine if the table load ~~factor~~ is below a preselected threshold. (Typical about 75% of the table) If the load ~~factor~~ is not below the threshold, the storage table is too full to be accessed efficiently, and box 77 is entered to report that the the table is full and the record cannot be inserted. The procedure then terminates in terminal box 75. If the load ~~factor~~ is below the threshold, box 78 is entered where the record to be inserted is placed in the empty cell position found by the search table procedure. In box 79, the load ~~factor~~ is adjusted to reflect the addition of one record to the storage table, the procedure reports that the record was inserted in box 80 and the procedure terminated in box 75.

The insert procedure illustrated generally in FIG. 5 is implemented in the Appendix as PASCAL-like pseudocode. Source code suitable for compilation and execution on any standard hardware and software computing

system can readily be devised from this pseudocode and the flowcharts of the FIG. 5 by any person of ordinary skill in the art.

In FIG. 6 there is show a detailed flowchart of a retrieve procedure which is used to retrieve a record from the data base 24 of FIG. 2. Starting in box 90, the search table procedure is invoked in box 91, using the key of the record to be retrieved as the search key. In box 92 it is determined if a record with a matching key was found by the search table procedure. If not, box 93 is entered to report failure of the retrieve procedure and the procedure is terminated in box 96. If a matching record was found, box 94 is entered to copy the matching record into a buffer store for processing, box 95 entered to return an indication of successful retrieval and the procedure terminated in box 96.

The pseudo-code for the retrieve procedure of FIG. 6 is included in the Appendix. Executable code for all common hardware and system software arrangements can readily be devised by those skilled in the art from the flowchart and the pseudo-code.

In FIG. 7 there is shown a detailed flowchart of a delete procedure useful for actively removing records from the data base 24 of FIG. 2. Starting at box 100, the procedure of FIG. 7 first invokes the search table procedure of FIG. 3 in box 101, using the key of the record to be deleted as the search key. In box 102, it is determined if the search table procedure was able to locate a record with a matching key. If not, box 103 is entered to report failure of the deletion procedure and the procedure is terminated in box 106. If a matching record was found, as determined by box 102, the remove procedure of FIG. 4 is invoked in box 104. As noted in connection with FIG. 4, this procedure removes the record to be deleted and, at the same time, closes the search chain. Box 105 is then entered to report successful deletion to the calling program and the procedure is

terminated in box 106.

The delete procedure illustrated generally in FIG. 7 is implemented in the Appendix as PASCAL-like pseudocode. Source code suitable for compilation and execution on any standard hardware and software computing system can readily be devised from this pseudocode and the flowchart of FIG. 7 by any person of ordinary skill in the art.

The attached Appendix contains pseudocode listings for all of the programmed functions necessary to implement a data base manager 23 (FIG. 2) operating in accordance with the present invention. These listings follow the flowcharts of FIGS. 3-7 and further explain and elucidate the flowcharts. Any person of ordinary skill in the art will have no difficulty implementing these functions in any desired program language to run on any desired computer hardware configuration.

It should also be clear to those skilled in the art that further embodiments of the present invention may be made by those skilled in the art without departing from the teachings of the present invention.

APPENDIX

Functions Provided

The following functions are made available to the application program:

insert (record: record type)

Returns replaced if a record associated with record.key was found in the table and subsequently replaced.

Returns inserted if a record associated with record.key was not found in the table and the passed record was subsequently inserted.

Returns full if a record associated with record.key was not found in the table and passed record could not be inserted because load factor has reached max load factor.

retrieve (record: record type)

Returns success if record associated with record.key was found in the table and assigned to record.

Returns failure if search was unsuccessful.

delete (record key: record key type)

Returns success if record associated with record key was found in the table and subsequently deleted.

Returns failure if none found.

Definitions

The following formal definitions are required for specifying the insertion, retrieval, and deletion algorithms:

```
const table size          /* size of hash table */

const max load factor     /* 0 ≤ max load factor < 1 */

var table: array[0 .. table size-1] of record type;
                  /* hash table */

var load: 0 .. table size-1;
                  /* number of occupied entries of
                     hash table array (initially 0) */
```

Algorithms

Algorithms for the functions described above are given below:

```
function insert (record: record type):
         (replaced, inserted, full );

  var position: 0 .. table size-1;
                /* position in table to update or
                   insert (returned by search table) */

  begin

    if search table (record.key, position)

       then begin

         table[position] := record;
         return (replaced)

       end

       else if load/table size < max load factor

         then begin

           load := load+1;
           table[position] := record;
           return (inserted)

         end

         else return (full)

  end                /* insert */
```

```
function retrieve (var record: record type): (success, failure);

  var position: 0 .. table size-1;
            /* position in table where record
               resides (returned by search table) */

5   begin

      if search table (record.key, position)

         then begin

           record := table[position];
10         return (success)

         end

         else return (failure)

      end  /* retrieve */
```

```
function delete (record key: record key type):
      (success, failure);

  var position: 0 .. table size-1;
            /* position in table where record
               resides (returned by search table) */

  dummy variable: 0 .. table size-1;
            /* last two arguments to remove are not relevant */

  begin

    if search table (record key, position)

      then begin

        remove (position, true, dummy variable,
              dummy variable);

        return (success)

      end

      else return (failure)

  end    /* delete */
```

```
function search table (record key: record key type;
        var position: 0 .. table size-1): boolean;

    /* search table for record key and delete expired
       expired records in target chain; position is set to
       index of found record or appropriate empty cell */

var i: 0 .. table size-1;
       /* used for scanning chain, both forwards & backwards */

     pos empty: 0 .. table size-1;
       /* index of leftmost empty cell to right of position */

     is rec found: boolean;
       /* indicates whether search is successful */


begin

  position := hash (record key);
  is rec found :=  false;

  if table[position] is not empty then

      begin

        i := position;      /* loop initialization */
        repeat              /* scan forward to end of chain
                               containing table[position]  */

            i := (i+1) mod table size

        until (table[i] is empty);

        pos empty := i;
        i := (i-1+table size) mod table size;
```

```
        while (table[i] is not empty) do
              /*scan chain in reverse,
                deleting expired entries */

            begin

5
              if table[i] is expired then
              remove (i, is rec found,
               position, pos empty)

                    else if table[i].key = record key
10
                                  then begin

                                      is rec found := true;
                                      position := i
15
                                  end;

              i := (i-1+table size)
              mod table size

            end;   /* while  */

20        if not is rec found then position := pos empty

        end;    /* then  */
      return (is rec found)
    end                    /* search table  */
```

there 2 should line up

```
procedure remove (cell to del:
 0 .. table size-1; is rec found: boolean;
 var pos of search rec, pos empty: 0 .. table size-1;

          /* Delete table[cell to del] */

 5   var i, j: 0 .. table size-1;

     begin

        load := load-1;

        do forever

10        i := cell to del;     /* save position of emptied slot */

          repeat                /* scan forward looking for a
                                   record to fill hole in chain */

             cell to del := (cell to del+1) mod table size;

             if table[cell to del] is empty

15              then begin

                   table[i] := empty;

                   if not is rec found then

                      if (pos of search rec ≤ i < pos empty)
20                    or (i< pos empty < pos of search rec)
                      or (pos empty< pos of search rec ≤
                      i) then pos empty := i;

                   return

                end;
```

```
            j := hash (table[cell to del].key)

        until (j ≤ i < cell to del)
        or (i < cell to del < j)
        or (cell to del< j ≤ i);

5       table[i] := table[cell to del];
            /* use table[cell to del] to plug hole in chain */

        if (is rec found) and
        (pos of search rec = cell to del)
        then pos of search rec := i

10      end

    end                     /* remove */
```

What is claimed is:

1. An information storage and retrieval system using hashing techniques to provide rapid access to the records of said system and utilizing a linear probing technique to store records with the same hash address, said system comprising

a record search means utilizing a search key to access a chain of records having the same hash address,

means for removing all expired records from said chain of records, and

means, utilizing said record search means, for inserting, retrieving and deleting records from said system.

2. The information storage and retrieval system according to claim 1 further comprising

means for recursively moving a record from a later position in said chain of records into the position of one of said expired records.

3. The information storage and retrieval system according to claim 1 further including

means for inhibiting the insertion of new records into said system when the available storage space falls below a preselected value.

4. The information storage and retrieval system according to claim 3 wherein

means for re-enabling the insertion of new records into said system when the available storage space rises above said preselected value.

5. An automatically decontaminating hashed storage table comprising

means for accessing said storage table for inserting, retrieving and deleting records, and

means for automatically removing expired records from said table each time said table is accessed.

6. A method for storing and retrieving information records using hashing techniques to provide rapid access to said records and utilizing a linear probing technique to store records with the same hash address, said system comprising

accessing a chain of records having the same hash address,

removing all expired records from said chain of records, and

utilizing said record search means, for inserting, retrieving and deleting records from said system.

7. The method according to claim 6 further comprising the step of

moving a record from a later position in said chain of records into the position of one of said expired records.

8. The method according to claim 6 further including the step of

inhibiting the insertion of new records into said system when the available storage space falls below a preselected value.

9. The method according to claim 8 further comprising the step of

re-enabling the insertion of new records into said system when the available storage space rises above said preselected value.

10. A method for automatically decontaminating a hashed storage table comprising the steps of

accessing said storage table for inserting, retrieving and deleting records, and

automatically removing expired records from said table each time said table is accessed.

FIG. 1

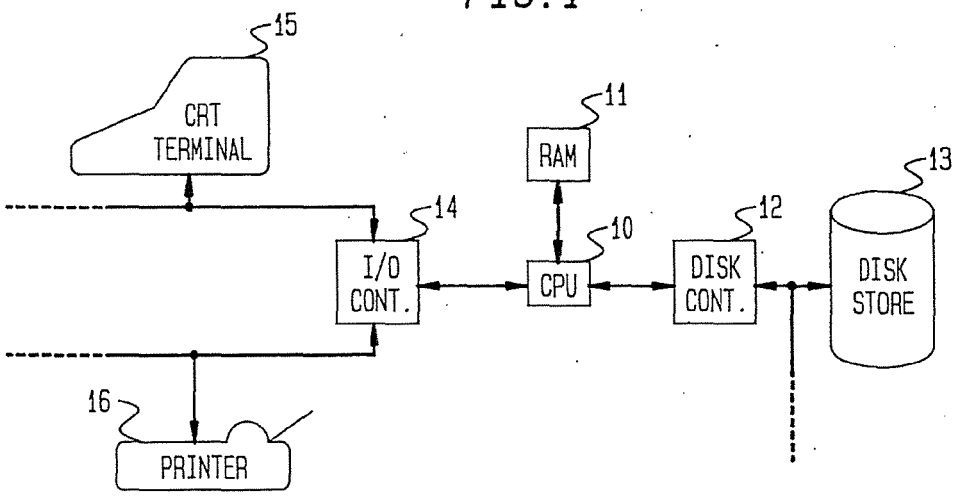
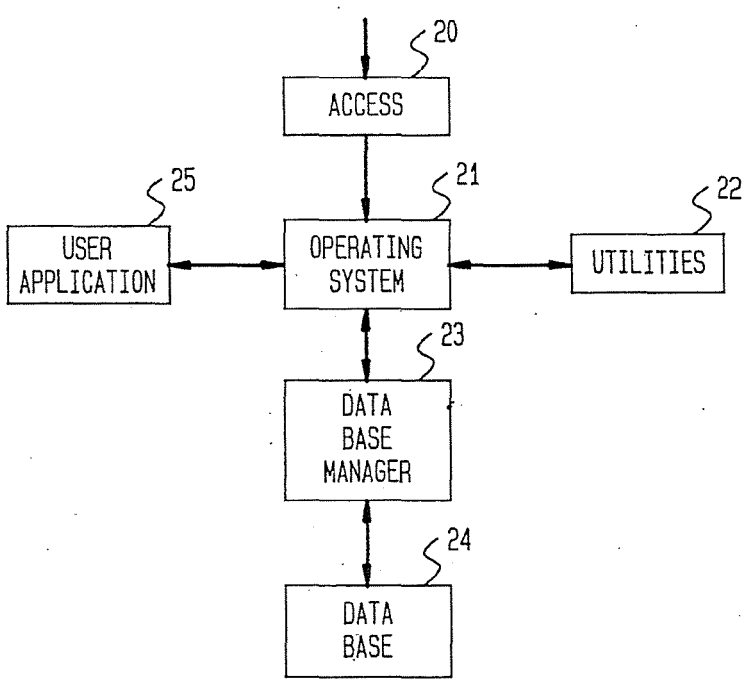

15
CRT
TERMINAL
11
RAM
13
14
12
10
I/O
CONT.
CPU
DISK
CONT.
DISK
STORE
16
PRINTER

FIG. 2


20
ACCESS
25
21
22
USER
APPLICATION
OPERATING
SYSTEM
UTILITIES
23
DATA
BASE
MANAGER
24
DATA
BASE

SEARCH-TABLE

FIG. 3

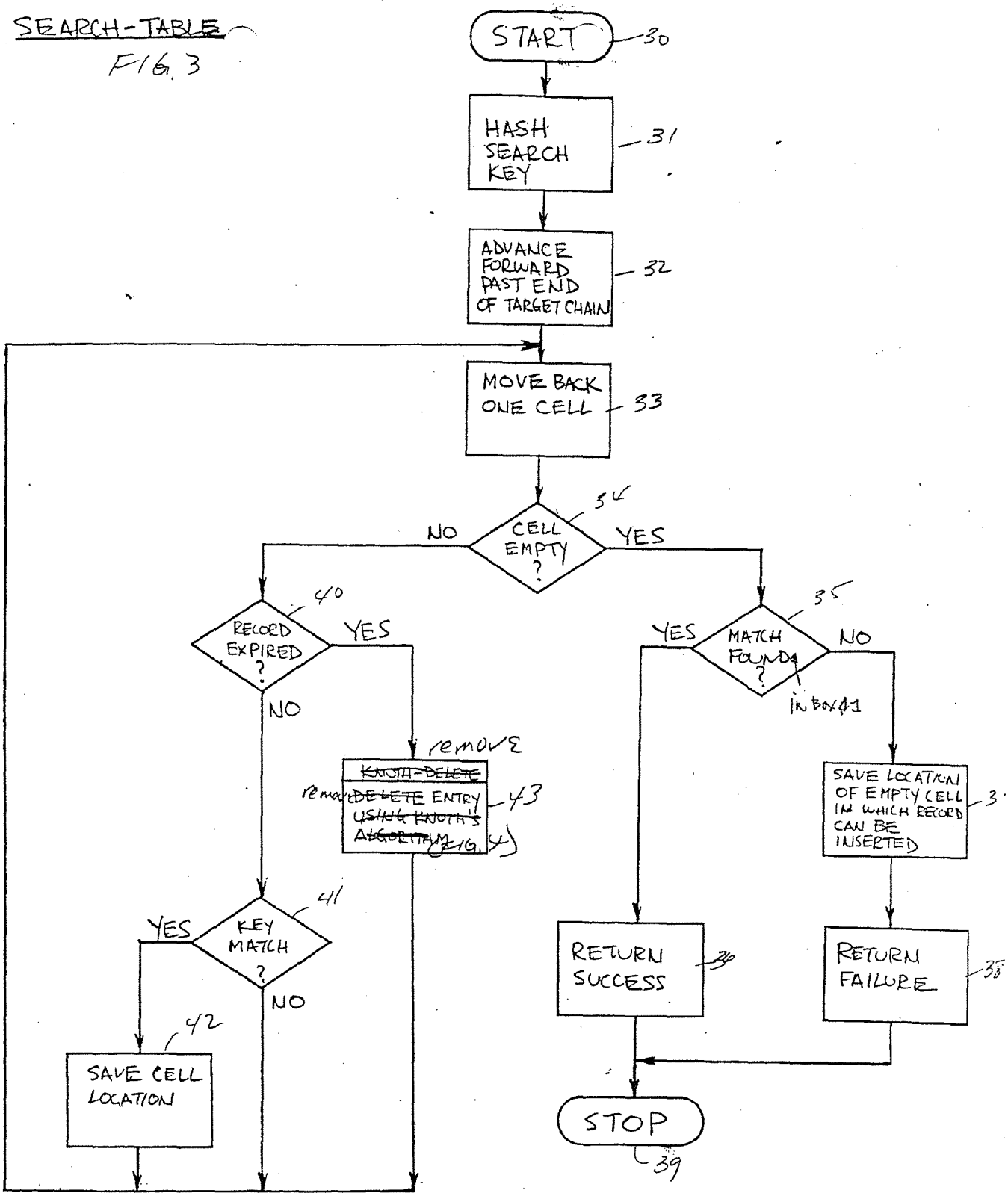
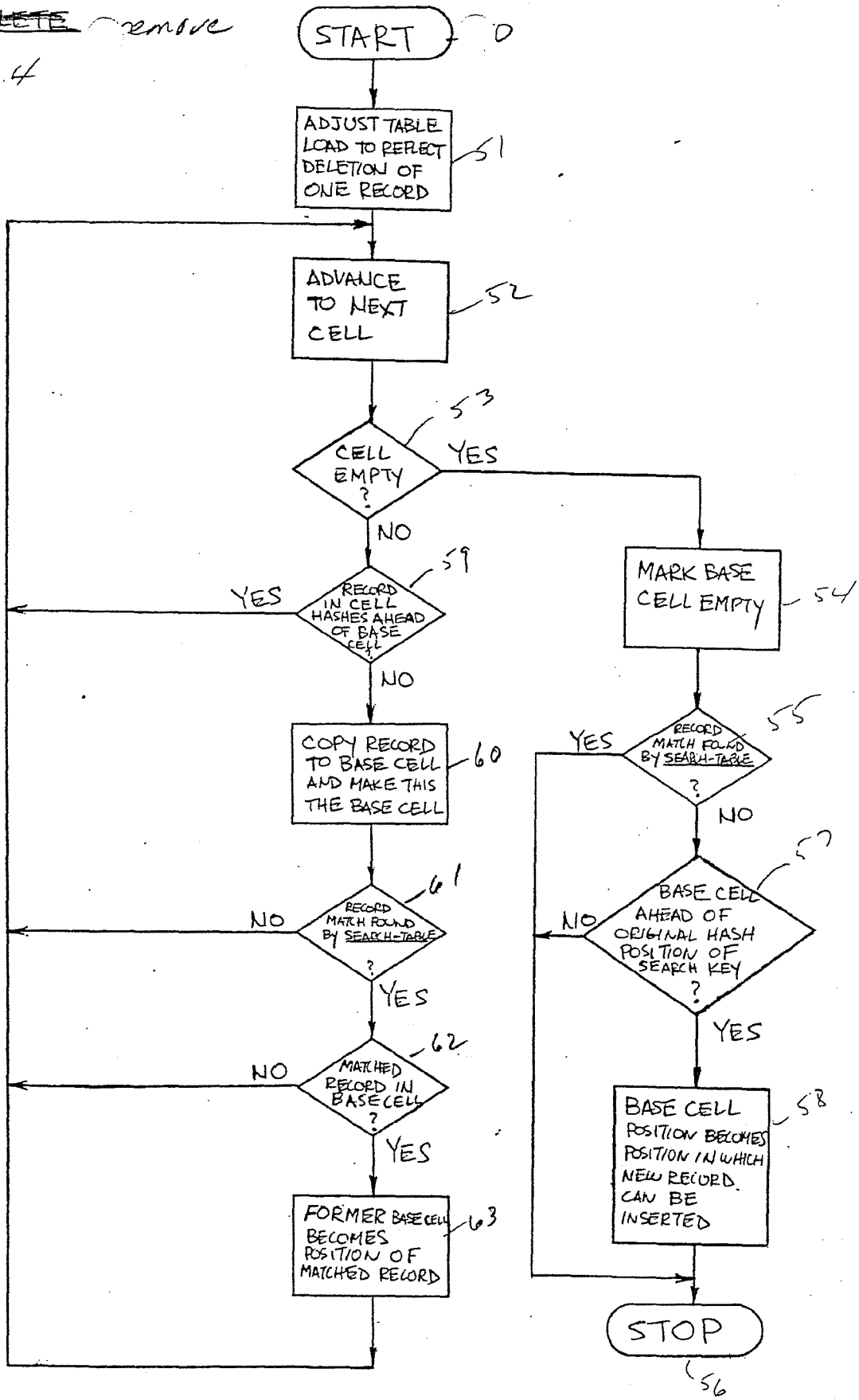


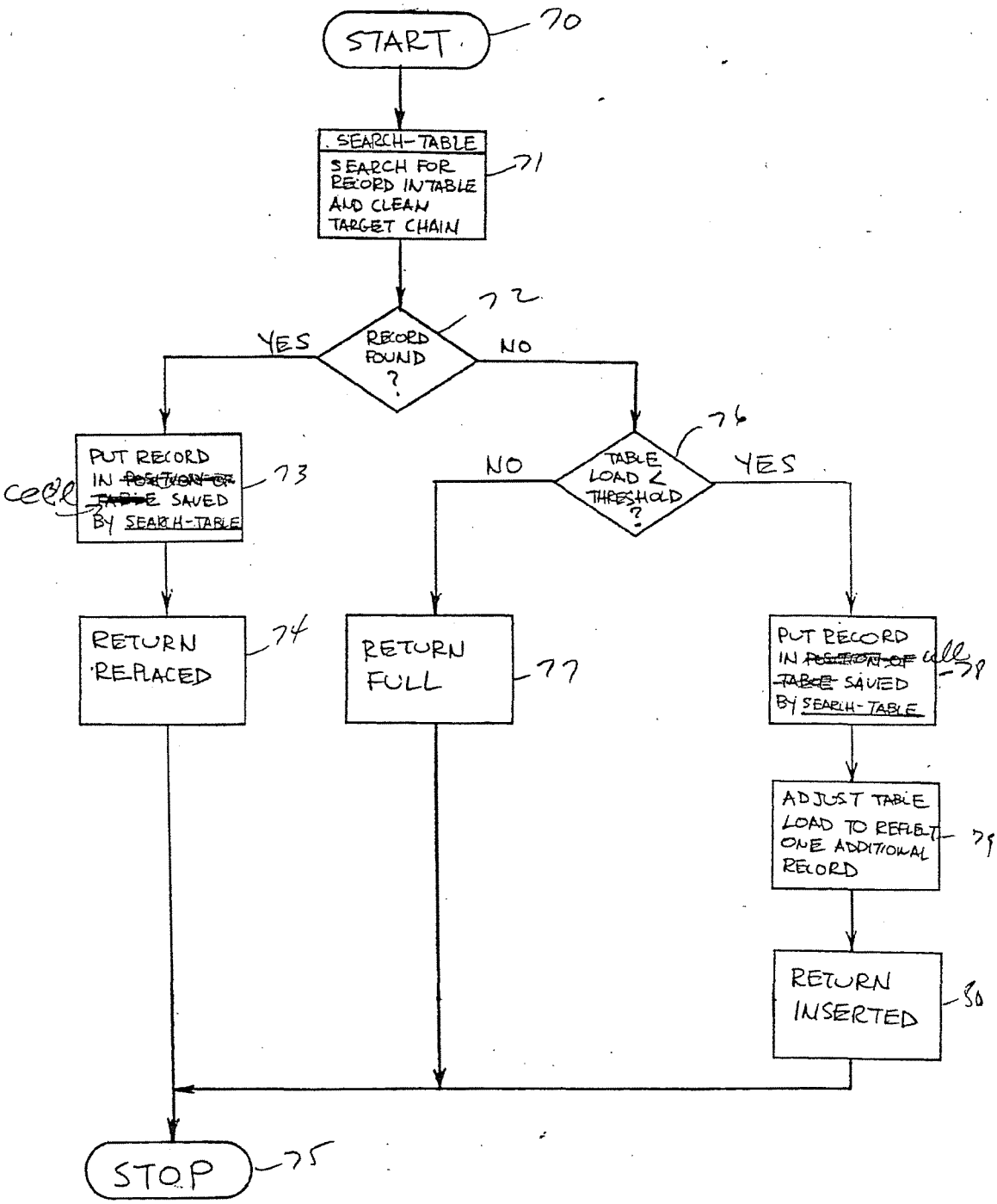
FIG. 2 (PRIOR ART)

KNUTH ~~DELETE~~ remove

FIG. 4

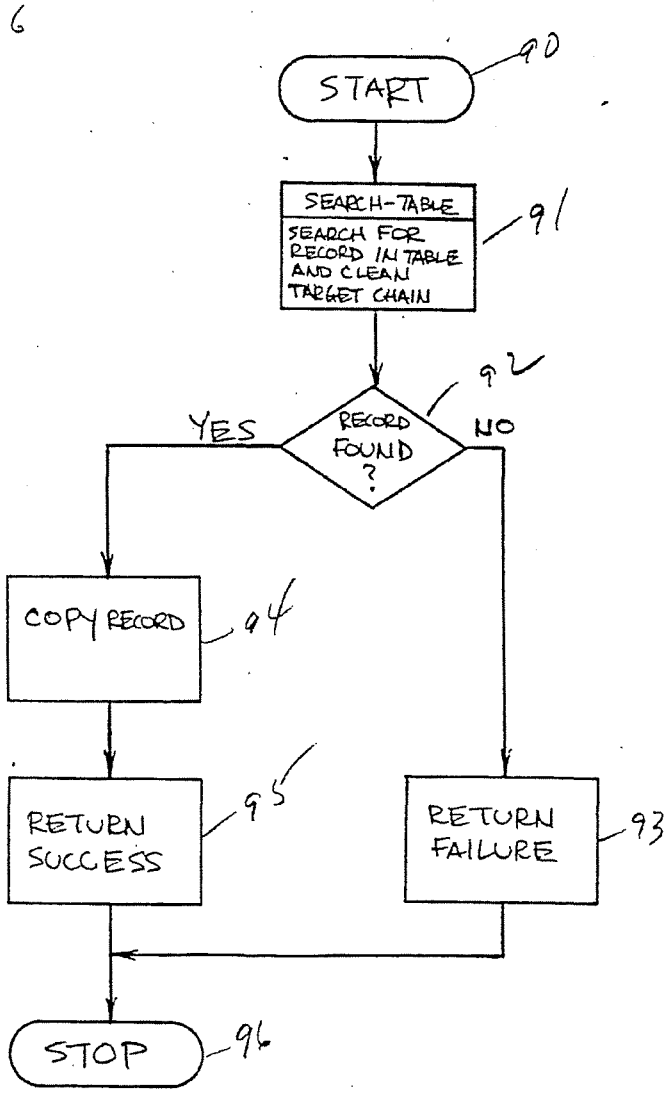


INSERT

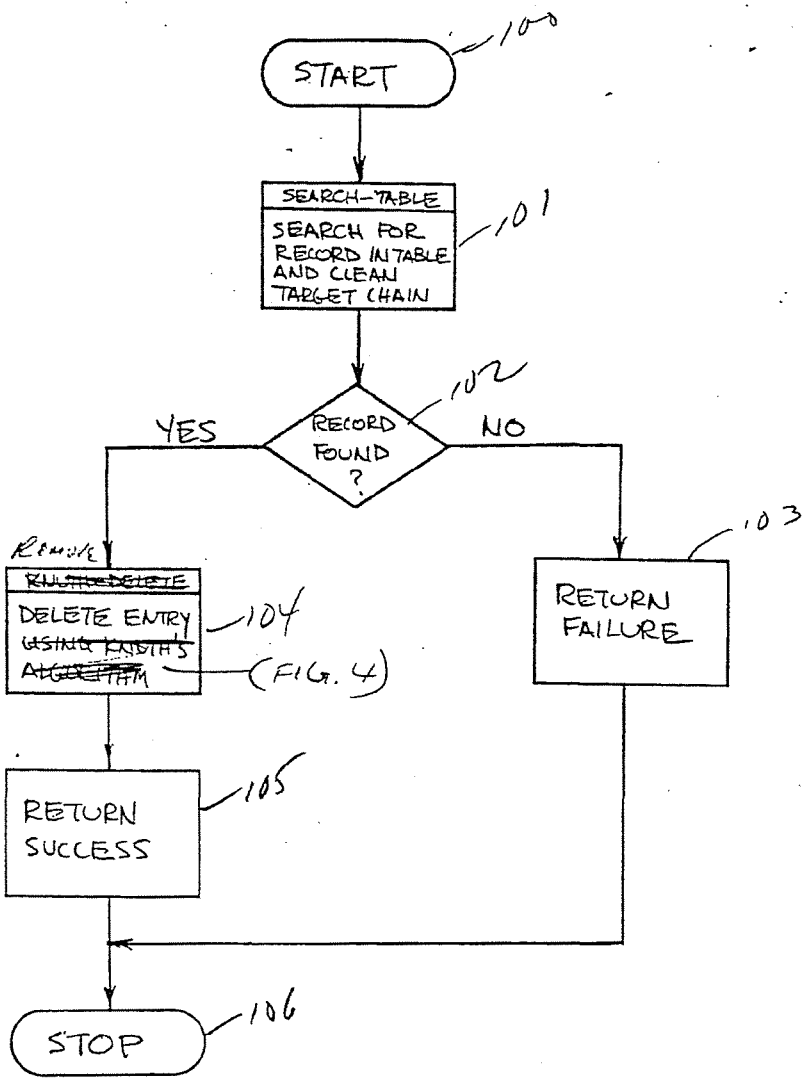

START
70
SEARCH-TABLE
SEARCH FOR RECORD IN TABLE AND CLEAN TARGET CHAIN
71
RECORD FOUND ?
72
YES
NO
PUT RECORD IN ~~POSITION OF~~ ~~TABLE~~ SAVED BY SEARCH-TABLE
73
cell
TABLE LOAD < THRESHOLD ?
76
NO
YES
RETURN REPLACED
74
RETURN FULL
77
PUT RECORD IN ~~POSITION OF~~ ~~TABLE~~ SAVED BY SEARCH-TABLE
78
cell
ADJUST TABLE LOAD TO REFLECT ONE ADDITIONAL RECORD
79
RETURN INSERTED
80
STOP
75

RETRIEVE

FIG. 6



DELETE FIG. 7


START
100
SEARCH-TABLE
SEARCH FOR RECORD IN TABLE AND CLEAN TARGET CHAIN
101
RECORD FOUND ?
102
YES
NO
DELETE ENTRY
104
(FIG. 4)
RETURN FAILURE
103
RETURN SUCCESS
105
STOP
106

PATENT AUTHORIZATION

App/245A (1)

INVENTOR(S) (Give full name, use no initials)	CASE NO.
Richard Michael Nemes (25655) 699-4365	2

TITLE OF INVENTION

Methods and Apparatus for Information Storage and Retrieval

PATENT JOB NO.	TYPE OF APPLICATION	FILING DATE	SERIAL NO.
693	New		

RELATED APPLICATIONS FOR PATENT

R. M. Nemes Case 1

ABSTRACT

Hashing is a well-known technique for rapid retrieval of data records by translating ('hashing ') the storage key into a storage location. Data with some form of time dependency expires after some given time and, if retained in the data storage mechanism, contaminates the storage space by blocking new data and requiring searches to be extended over the now-expired data records.

The present invention improves the performance of hashed data storage systems with time dependent data by removing such data records in the process of subsequent data insertions, retrievals or deletions. More particularly, for each access of the data base, the entire chain of data records attached to the accessed data record is searched for expired data records and such expired records are deleted at that time. Such automatic decontamination of the data base allows rapid hashing access without the penalty of eventual debilitating contamination with expired but undeleted records.

SIGNED	DATE	SIGNED	DATE
J. T. Peoples	12-31-87		

PT 794 3-84

Bell Communications Research
Intellect Property Matters

PRELIMINARY RECORD OF THE INVENTION

Inventor-Case No.
R. M. NEMES CASE 8

When did you formulate at least an outline of the invention (give approximate date if unable to fix exactly):

Date: JULY 10, 1986

Background circumstances of the invention:

INVENTOR WAS GIVEN THE ASSIGNMENT OF DIVISING AN EFFICIENT TECHNIQUE FOR STORING AND RETRIEVING CALLING CARD FRAUD DATA IN THE SERVICE CONTROL POINT (SCP) LINE INFORMATION DATA BASE (LIDB) SYSTEM.

Date first disclosed to anyone other than a co-inventor:
Name(s): EDWIN MILLER
JULY, 1986

Where is first recorded description:
"DETAILED DESIGN SPECIFICATIONS FOR LIDB ON-LINE FRAUD CONTROL FACILITIES,"
Date: JULY 10, 1986 DS-LIDBO.2-LBDU-Sa

Where is first sketch or drawing:
"DETAILED DESIGN SPECIFICATIONS FOR LIDB ON-LINE FRAUD CONTROL FACILITIES," DS-LIDBO.2-LBDU-Soo.
Date: JULY 10, 1986

Where is first recorded description, sketch or drawing witnessed by one other than a co-inventor:
"DETAILED DESIGN SPECIFICATIONS FOR LIDB ON-LINE FRAUD CONTROL FACILITIES," DS-LIDBO.2-LBDU-Sa
Date: JULY 10, 1986

Name(s) of witness(es):
EDWIN MILLER

Subsequent early records which pertain to the invention (indicate nature and location): BELLCORE MEMORANDA LOCATED IN PROJECT LIBRARY OF DISTRICT 25614:

1) "DETAILED DESIGN SPECIFICATIONS FOR LIDB ON-LINE FRAUD CONTROL FACILITIES," DS-LIDBO.2-LBDU-S003, JULY 10, 1986.

2) "DETAILED DESIGN SPECIFICATIONS FOR LIDB FRAUD PROCESS SYNCHRONIZATION AND COMMUNICATION," DS-LIDBO.2-LBDU-S004, JULY 31, 1986.

Pt. 500 2-85

(Continue on reverse side)

What was constructed, prepared or assembled to demonstrate the invention:

SOFTWARE PROGRAMS COMPRISING THE SERVICE CONTROL POINT (SCP) LINE INFORMATION DATA BASE (LIDB) FRAUD CONTROL SYSTEM.

Who performed this construction, preparation or assembly:

RICHARD NEMES

Date: JANUARY, 1987

What records are there of the construction, preparation or assembly:

NONE

Who performed tests which demonstrated principles of the invention:

LORI VINCIGUERRA

Date: FEBRUARY, 1987

Names of test witnesses other than co-inventors:

EDWIN MILLER
LORI VINCIGUERRA
MARY KENNEY

Identify records of these tests:

MEMORANDA IN PROJECT LIBRARY OF DISTRICT 2561.

Identify others who performed tests:

EDWIN MILLER PATRICIA HAWKINS
LORI VINCIGUERRA

Date: —

Identify records of these other tests:

FIELD TESTS IN THE FILES OF PATRICIA HAWKINS

Inventor(s) signature(s)	Date:
Richard M. Nemes	JAN 7, 1988

ASSIGNMENT AND AGREEMENT

For value received, I, Richard Michael Nemes

of Brooklyn, in the County of Kings, and State of New York

hereby sell, assign and transfer to **BELL COMMUNICATIONS RESEARCH, INC.**, a corporation of the State of Delaware, having an office at 290 West Mount Pleasant Avenue, Livingston, New Jersey 07039, U.S.A., and its successors, assigns and legal representatives, hereinafter collectively **BELL COMMUNICATIONS RESEARCH**, the entire right, title, and interest in and to certain inventions related to

Methods and Apparatus for Information Storage and Retrieval

described in an application for Letters Patent of the United States, executed by me of even date herewith, including all rights of priority arising from the application aforesaid, and all the rights and privileges in said application and under any and all forms of protection, including Letters Patent, that may be granted for said inventions in the United States and any countries foreign to the United States.

I authorize **BELL COMMUNICATIONS RESEARCH** to make application for such protection in its own name and maintain such protection in any and all countries foreign to the United States, and to invoke and claim for any application for patent or other form of protection for said inventions, without further authorization from me, any and all benefits, including the right of priority provided by any and all treaties, conventions, or agreements.

I hereby consent that a copy of this assignment shall be deemed a full legal and formal equivalent of any document which may be required in any country in proof of the right of **BELL COMMUNICATIONS RESEARCH** to apply for patent or other form of protection for said inventions and to claim the aforesaid benefit of the right of priority.

I request that any and all patents for said inventions be issued to **BELL COMMUNICATIONS RESEARCH** in the United States and in all countries foreign to the United States, or to such nominees as it may designate.

I agree that, when requested, I shall, without charge to **BELL COMMUNICATIONS RESEARCH** but at its expense, sign all papers, take all rightful oaths, and do all acts which may be necessary,

desirable or convenient in connection with said applications, patents, or other forms of protection.

Richard Michael Nemes

Date: 1/28/88

United States of America)
)
State of New Jersey) ss.:
)
County of Middlesex)

On this 28th day of January, 1988,

before me personally came Richard Michael Nemes

to me known to be the individual described in and who executed the foregoing instrument, and acknowledged that he executed the same.

Christine K. Captain
Notary Public

CHRISTINE K. CAPTAIN
NOTARY PUBLIC OF NEW JERSEY
My Commission Expires 4/2/92

RECORDED
PATENT & TRADEMARK OFFICE
FEB -2 1988

COMMISSIONER OF PATENTS
AND TRADEMARKS OFFICE

REEL 4823 FRAME 0265

1J-A 5-84