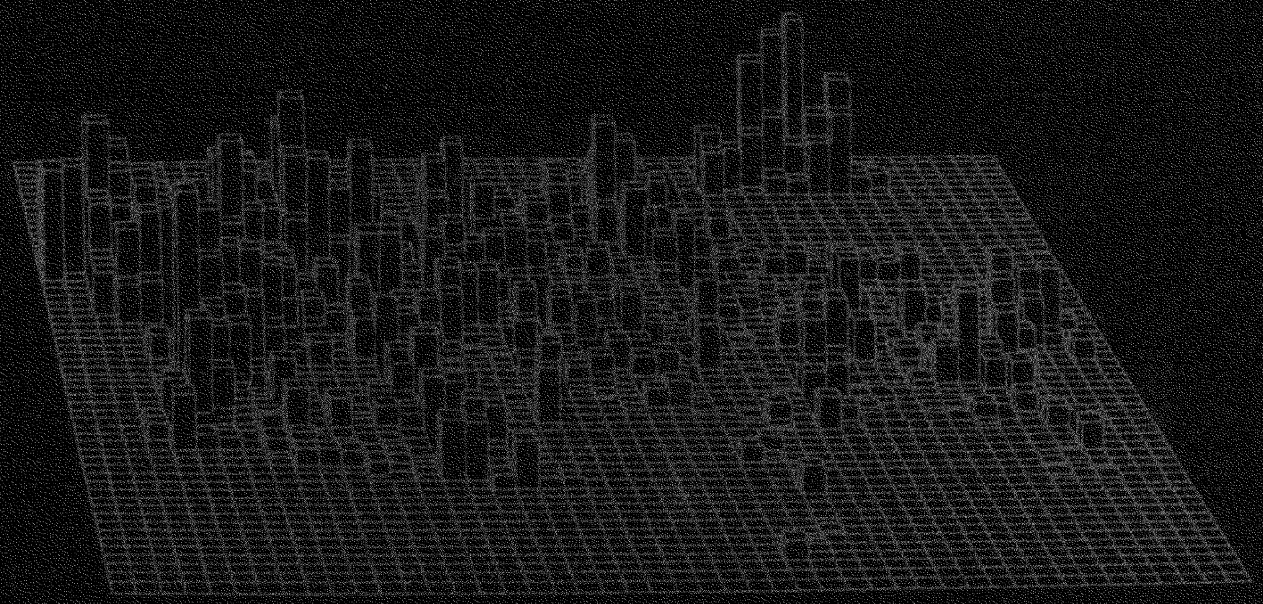
# CRU

# A locational index for the 1971 kilometre-square population census data for Great Britain

by M. Visvalingam



Census Research Unit
Department of Geography
University of Durham

Working paper 12

The Census Research Unit, Department of Geography, University of Durham, is a small group of research workers investigating aspects of the theory and use of census data. It is currently funded as a research project by the Social Science Research Council.

The diagram on the cover represents total population per 1 km grid square in the northern part of County Durham: the height of each column is proportional to the population in that square. The county is viewed from the west, Gateshead being at the extreme left margin, West Hartlepool at the far right and Bishop Auckland at the centre-right. The original surface was calculated and drawn by computer.

UNIVERSITY OF DURHAM

DEPARTMENT OF GEOGRAPHY

CENSUS RESEARCH UNIT

WORKING PAPER No. 12

OCTOBER 1977

# A LOCATIONAL INDEX FOR THE 1971 KILOMETRE-SQUARE POPULATION CENSUS DATA FOR GREAT BRITAIN

M. VISVALINGAM

CONTENTS

# A LOCATIONAL INDEX FOR THE 1971 KILOMETRE SQUARE POPULATION CENSUS DATA FOR GREAT BRITAIN


## Summary

A primary index was designed to facilitate the location of census records for one-kilometre square areas within Great Britain from four census files. This paper describes the structure and performance of the indexing system, which is applicable to any stable grid-based data set which does not present update problems. The primary index is a compactly encoded, two-level, nondense index, with concatenated keys and pointers to four separate files.


## 1. INTRODUCTION

The Census Research Unit (CRU) of the Geography Department, University of Durham is working primarily with the 1971 population census data, made available in aggregated form for 152,440 one-kilometre squares in Great Britain. The automated production of a census atlas of Great Britain involves the use of data available for as many locations as possible. However, other research objectives, such as the identification of demographic types in Britain, the study of urban deprivation or the evaluation of the effects of scale, would require a subset of the data records. Moreover, non-CRU users may be interested in data for only a selected sub-area of Britain.

The primary index facilities the extraction of data records for a sub-area. The spatial entity has to be expressed in terms of a list of x and y co-ordinates which form the primary keys of the one-kilometre grid squares involved. A forthcoming paper (Rhind and Visvalingam) describes the suite of routines which convert user-specified spatial entities to the list of 100 metre references. The design of the locational index is conditioned by the properties of the data base and by the operating characteristics of NUMAC (Northumbrian Universities Multiple Access Computer). The CRU has access to an IBM 370 computer, functioning under the Michigan Terminal System (MTS). The MTS file system does not permit users to locate records on specific areas of a disc pack. Hence the CRU can only ensure that the records are placed in physical sequence by writing the files one at a time, sequentially

onto the disc pack. MTS stores and retrieves logical records (in variable length spanned format), in one-page (4096 bytes) physical blocks. The MTS user has access only to the relative record numbers used by the MTS file system; thus the indexing system is a logical design built on top of the MTS file system.

## 2. CHARACTERISTICS OF THE DATA BASE

### 2.1. Favourable Characteristics

#### 2.1.1. Read only access

The census data, once compacted and stored,are only accessed for reading purposes. Hence the file structure for indexing need not consider the problems of update.

#### 2.1.2. Unique spatial keys

The Office of Population Censuses and Surveys provides census statistics in two files, namely the 100% and 10% Small Area Statistics. Each of these files contains pairs of records for each populated kilometre square in Britain. In the CRU system (Visvalingam and Perry, 1976) all four record types are stored in a highly compacted form in separate files to minimise the time needed to read any one record type. Hence each record within a file possesses a unique spatial key.

#### 2.1.3. Records sorted by spatial key

The CRU data files are sorted by their spatial keys so that records occur in decreasing value of northings (Y co-ordinate) then increasing value of eastings (X co-ordinate); i.e. data records are placed to occur in west to east kilometre strips, starting in northern Scotland and progressing southwards. Hence, the primary locational index need not contain an entry for each stored record (Engles, 1972); rather a single entry locates the group of records with a given Y co-ordinate. The X- co-ordinate is found by short sequential scan. Such an index is referred to as 'nondense' (Wagner, 1973).

### 2.1.4. Limited function of multivariate files

The multivariate record files, to which the primary index points, perform limited functions. They are generally read only for three purposes, namely the derivation of functional variables, the preparation of data for statistical packages and/or the extraction of data for sub-areas. Most of the other processing options, such as mapping and statistical analysis and the aggregation of data for larger spatial units, access the derived variables (Rhind et al,1977). However, the same locational index can be used to identify data elements in univariate lists.

## 2.2. Unfavourable Characteristics

### 2.2.1. Records of varying length

All data records are stored in a highly compacted form, which results in the records varying in length from 8 to 954 bytes. Hence the location and direct access of records by the calculation method is difficult.

### 2.2.2. Uneven spatial distribution of data locations

As data are provided only for the populated kilometre squares in Great Britain, each data record includes a 4-byte spatial key, which should easily be separated into its X and Y National Grid References. The indexing scheme thus needs to identify the locations for which data are/are not available.

### 2.2.3. Irregular presence of data types for each location

For each populated kilometre square there may be as many as four categories of data (see above). These are :

A - 100% sample statistics on 471 population characteristics
B - 100% sample statistics on 449 household characteristics
C - one set of 10% sample statistics on 368 socio-economic characteristics
D - another set of 10% sample statistics on a further 283 socio-economic characteristics

There was some discrepancy between the 100% and 10% files not only in terms of the length of records but also in terms of the number of records. A and B were provided for 152,440 one-kilometre squares, while C and D were only available for 87,975 one-kilometre squares. Of these, only 147,685 population records, 147,408 household records and 54,464 10% records contained non-zero values (see Visvalingam and Perry, 1976).

2.2.4. Different suppression criteria for different categories of data

The content of the above types of records also varies, depending upon suppression of data, owing to confidentiality restraints. The criteria for suppression and their effects are as follows :

A If less than 25 people reside in the kilometre square, A is suppressed and data are only available for the total number of people, the total number of males and the total number of females residing in the kilometre square. Only 67,546 of the 152,440 population records were unsuppressed.

B If there are fewer than 8 households in a kilometre square, than the only item of data available in B is a count of the total number of households in the square and the rest of the data on housing are suppressed. Only 68,421 of the 152,440 household records are unsuppressed.

C and D All data on socio-economic variables are suppressed if the 10% sample includes only one private household. Of the 87,975 10% records provided, only 54,464 records of C and 54,153 of D contained useful data.

As the criteria for suppression of the various record types are only indirectly related, unsuppressed data for the above three categories exist for non-identical subsets of kilometre squares within Britain. Table 1 gives the union, intersection and difference of the above data sets.

TABLE 1 : SOME RELATIONSHIPS BETWEEN THE UNSUPPRESSED DATA SETS

A) Number of one-kilometre squares with unsuppressed data in one or other of the files

| | Population | Household |
|---|---|---|
| Household | 70,947 | |
| 10% | 70,025 | 71,001 |

B) Number of kilometre squares with unsuppressed data in both files

| | Population | Household |
|---|---|---|
| Household | 65,021 | |
| 10% | 51,985 | 51,885 |

C) Number of kilometre squares with unsuppressed data in only one of the two files

| | Population | Household |
|---|---|---|
| Household | 5,926 | |
| 10% | 18,040 | 19,116 |

2.2.5. Data adjustments for purposes of confidentiality

To ensure confidentiality of the data, an error component is deliberately introduced by OPCS within the unsuppressed data counts of A and B (the 100% statistics). This process consists of the addition of +1, Ø, or -1 to all data counts. As a result, those items of data which were derived by the accumulation of other primary data items contain a larger component of error. For example, figures of 17 and 18 people were recorded for the resident population in unsuppressed 100% population records, while counts of two households were found in unsuppressed 100% household records. There are 1,245 100% population records with the total population adjusted to below 25 people in the kilometre square and 2,634 household records with the number of households below eight. Hence the suppression status of data records could not be ascertained purely by the stipulated suppression criteria (see Visvalingam and Perry, 1976). During the storage process, other procedures were adopted for checking the suppression status and this was noted within the record header.

Although this feature does not directly affect the indexing system, it had to be considered at a very early stage in the design of the storage and retrieval system, so that indexes could be constructed on data other than those quoted by OPCS to indicate suppression.

## 3. FUNCTION AND CHARACTERISTICS OF THE PRIMARY LOCATIONAL INDEX

### 3.1. Function of the Primary Index

Engles (1972) defined the primary index as "a map which relates entity identifiers to the storage locations of their stored records..." The locational index is intended for the retrieval of data records (for a relatively small number of kilometre squares) by their spatial identifiers, namely their X and Y National Grid co-ordinates. A spatial subset may be described in numerous ways, for example by a set of point references for a random or dispersed scatter of kilometre squares or by a list of grid co-ordinates describing the outlines of one or more regular or irregular polygons enclosing a group of kilometre squares. The may also be described by circular areas, bands along paths or as features described by some other criterion (Baxter, 1976). The locational index expects these to be reduced to a list of X and Y co-ordinates for each kilometre square. However, it also permits selection of records by suppression status, as this is expedient for the derivation of variables from multiple record types.

The primary index cannot be used directly for the retrieval of area subsets described by non-spatial criteria. For example, it cannot retrieve URBAN areas defined to be those kilometre squares with more than N residents. However, quick indexes for the retrieval of records by their attributes or values may themselves access the locational index. This is especially useful for the location of records for the union, intersection or difference of area or attribute subsets (to be discussed in a separate paper).

The locational index also permits a check on the existence or non-existence of a record, by type, suppression status and location, without the necessity of reading the data file. A data file therefore is only accessed when the desired record is known to exist.

## 3.2. Characteristics of the Primary Index

### 3.2.1. Independent index

All indexing information is divorced from the data files, so that changes to the structure or design of the former do not affect an otherwise stable data base. Also the information is located on a different disc pack and hence the access of indexing information does not incur the movement of the disc arm of the data file.

### 3.2.2. Nondense index

The primary index need not contain an entry for each stored record if the stored records are in sequence by the collating value of their entity identifiers (Engles, 1972; Wagner, 1973). It can have one entry for a group of records. The record order (see 2.1.3 above) implies that, within a group of records with the same Y co-ordinates (hereafter called a Y partition),the X co-ordinate will increase in value. Thus only the pointers to the Y partitions need be stored. The nondense structure is especially valuable considering that there are over 152,000 unique spatial keys in each of the two 100% files.

### 3.2.3. Datamaps

Although the record with the specified X co-ordinate could be found by a short sequential scan within the partition, it was expedient to ascertain the existence of data for the given location from bitmaps (Rhind, 1974) before commencing search. Each Y partition that existed in the data file could be recorded in two primary bitmaps, one each for the 100% and 10% data files. The suppression status of these was similarly recorded in three secondary bitmaps, one each for indexing the population and household files and another for indexing the two 10% files. Y partition maps may exist only for the 100% files and not for the 10% files. Thus the locational index also maintains data maps of suppression relationships between the record types. These maps are easily compared to derive spatial subsets of interest, to be located via the pointers and primary bitmaps. Pointers locate the start of a partition and the primary bitmap indicates the location of the record in terms of an offset or displacement of records.

### 3.2.4. Concatenated pointers and bitmaps

Indexing information relating to the four data files is all stored within a single locational index. This minimises storage overheads and access time when more than one record type is desired. Wagner (1973) has already discussed the merits and disadvantages of concatenated keys and pointers.

### 3.2.5. Two-level index

The locational index consists of two levels. The high level index marks, with a 4-byte coded element, the existence or non-existence of a Y partition and, if the latter does exist, whether it exists in both 100% and 10% files or only in the former. The same element points to the location of further indexing information if search is to continue.

The sub-index, referenced by the high-level index, points to the start of the Y partitions in the four (or two)files. It also contains a count of the maximum number of one-kilometre squares for which data were provided by OPCS, the arrays of bitmaps, and information to index the bitmaps, namely the length of the bitmaps (standardised for each Y partition for ease of logical comparisons) and the X co-ordinates for the first and last elements (bits) of the bitmaps.

## 4. STRUCTURE OF THE PRIMARY INDEX

### 4.1. High Level Index

#### 4.1.1. Functions

These are :

(a) to determine the existence of records with specified Y co-ordinates in all files or just the 100% files.

(b) to index a sub-index if records do exist for the specified Y value.

#### 4.1.2. Structure

An array of 1,212 elements of 4 bytes each, corresponding to Y co-ordinates in the range 80 to 12,190 inclusive. The length of the elements was determined by the length of POINTERS to MTS sequential files.

4.1.3. Content

Ø - no data for Y strip

-ve - partition only in 100% files

+ve - partition in all files

IABS (non zero value) - pointer to start of a corresponding record in sub-index

The core requirements of the high level index were thus kept to a minimum by encoding three alternative types of information within the same array.

4.1.4. Location

The high level index is stored as the first record in a sequential file. The first time a Y- co-ordinate in the range 80 to 12,190 inclusive is encountered, the record is read into an array (IY) declared within the indexing routine, where it remains for the duration of the run. Data relevant to the specified Y co-ordinate is directly accessed in core by the simple mapping function, IY((Y-MINY)/10+1).

4.1.5. Necessity

The use of the high-level index for determining the existence of Y-partitions is an incidental though fortuitous benefit because, of the 1,212 elements, only 56 have zero and only 34 possess negative values. Hence the probability of search terminating at this level is small. The high level index is essential for locating the relevant records in the sub-index, since these records vary in length, owing to the variable lengths and dimensions of the arrays of bitmaps.

4.2. The Sub-Index

4.2.1. Functions

The sub-index

(a) points to the start of the Y partitions in the four data files;

(b) indicates the existence of data records (with the specified X co-ordinates) within the partition in each of the relevant files;

(c) indicates the location of existing records as a displacement (reckoned in numbers of records) from the start of the Y partition;

(d) indicates the suppression status of the above records.

4.2.2. Structure and content

The sub-index consists of a set of records, one for each existing Y partition. The records are of varying length and possess a complex structure, which consists of the following sub-structures and elements :

ND - a (2-byte) integer count of the maximum number of data records in the Y partition

NW - a (2-byte) integer value of the length of each bitmap (in 4-byte words)

MINX - the (2-byte) X co-ordinate value of the first bit in the bitmap, i.e. the first record in the partition

MAXX - the (2-byte) X co-ordinate value of the last record in the Y partition

POINTER(4) - an array of four (4-byte) integer elements, which contain the MTS 'pointers' to the Y partitions in the four data files. Each pointer, in turn, is composed of a 2-byte page (or physical record) number within the file and a 2-byte count of the offset (in bytes) within the page.

These pointers can be used by the sequential file system of MTS to commence reading records from the indexed position within the continuous sequence.

BITMAP (NW,I) - an array of I (five or three) bitmaps, each of NW (4-byte) elements. The first and fourth bitmaps are primary ones (see 3.2.3 above) corresponding to the 100% and 10% files respectively. The second, third and fifth bitmaps are secondary ones which have bits set for locations for which unsuppressed records are available in the population, household and both 10% files respectively. This arrangement of subscripts (rows) was designed to permit the omission of the last two rows if the Y partition does not exist in the 10% files. However the trimming was not executed, since the savings in storage were not large enough to justify the additional complexity in programming.

4.2.3. Location

The sub-index is stored with the high level index in the same sequential file. To be consistent with the data files, the records of the sub-index are themselves placed in decreasing Y-value sequence. A sequential file was chosen for several reasons. MTS offers the user only two types of file organisation, the sequential and line file organisations (see MTS Volume 1). For single records, the storage overheads of the line file are slightly smaller than with the sequential file. The line file directory requires 8 bytes per record (of system storage). While sequential files require only 6 bytes of system storage per stored record, indexed operations require a further 4 bytes of user storage for pointers to indexed records, bringing the total overhead to 10 bytes. However, line file records are restricted to a maximum of 255 bytes, whereas sequential files may have records up to 32,767 bytes long.

The MTS file system uses a line directory and a table look-up process for locating lines or records corresponding the specified line numbers, which can be formed from the X and Y co-ordinates. The mantenance of general purpose line directory blocks is inefficient in a large static file, since a pointer structure can be constructed by the user to locate more efficiently the required records in a sequential file.

Thus, for the existing keys, the location of the sub-index in a sequential file and indexing the records via pointers in a 4,848-byte area defined within the indexing routine was likely to be more efficient. Not only is the direct indexing of an array more efficient than table look-up, but the high-level index is also likely to be paged-in when the indexing routine is accessed. The MTS sequential file organisation is also preferable, as sub-index records can exceed 255 bytes for southern England, where Y strips contain data for more than 390 locations. However, decisions regarding the choice of file structures may need to be reconsidered when MTS distribution 4 becomes the chief operating system some time in 1978. In this system, restriction of line files to short records will be removed and other modifications to the file system are anticipated (personal communication, R.E. Vine).

4.2.4. Size of the sub-index

| | | |
|---|---|---|
| Number of records | : | 1,156 |
| Minimum length of record | : | 44 bytes |
| Maximum length of record | : | 324 bytes |
| Storage of arrays of pointers | : | 18,496 bytes |
| Storage for arrays of bitmaps | : | 175,700 bytes |
| Overheads of file organisation | : | 6,952 bytes |
| Storage of other information | : | 9,248 bytes |
| Total size of index file | : | 210,396 (52 pages) |

The storage requirements of the primary index were pruned in several ways. The datamaps were bit encoded; the length of each type of element within the primary index was kept to a minimum; the information content of the high level elements was maximised and system storage overheads were reduced by concatenating pointers and bitmaps for the four record types within the same index record.

The bitmaps are stored in higher units of 4 bytes (words) so that logical functions can be directly employed without intermediate processing. For the same reason, all bitmaps for a given partition were standardised and aligned although this involved the storage of redundant leading and trailing bits, especially in the secondary bitmaps.

On average, a page of index covers 367 pages of data or 9,246 data records; and if necessary these ratios can be further improved. As the 1,156 sub-index records are held in 52 pages, on average a page contains 22 to 23 records of the sub-index. As sub-index records vary in length from 44 to 324 bytes, one page transfer would include at least 12 such records.

## 5. PERFORMANCE OF THE PRIMARY INDEX

The primary index is just another data set and, especially at the level of the sub-index, it does not restrict retrieval procedures to any single strategy. Performance, as evaluated by empirical statistics, is partly dependent on the strategy adopted by the retrieval procedure and the efficiency of retrieval algorithms. However, it is possible to evaluate conditions under which bitmap indexing can be

either advantageous or wasteful compared with partitioned sequential scans.

## 5.1. Index for Partitioned Scans

Sequential scans within partitions also require pointers to the start of partitions in the four files. Moreover, the storage of the X co-ordinates, marking the extremities of each partition (i.e. MINX and MAXX) in the four files, would greatly facilitate the elimination of non-existent keys. These involve a total storage of 32 x 1156 bytes (12 pages) and are stored in a line file.

## 5.2. Implications of Retrieval via Bitmap Indexing and Sequential Scans

The retrieval of data via each of the two indexes has several features in common. Both involve the transfer of a minimum of one page of index record and an identical number of pages of data records, except when the last (or the last few) keys to be found in a partition are non-existent. This is because the datamaps of the sub-index permit the location of the relevant record within the partition only as an offset, in records, from a start address. The intervening records are passed over by the retrieval algorithm by means of the MTS routine, SKIP. As the record headers of MTS sequental files do not include the number of the logical record, it appears that the MTS, SKIP operation is a serial process, involving the transfer of all intermediate MTS indexing information in a chain of data set control blocks (personal communication, R.E. Vine). Thus, in terms of the data retrieval time (including the access or seek time, rotational delay, and data transfer time), the performance of both methods ought to be very similar. However, when a significant number of high X-value keys within partitions are non-existent, the partitioned scans may involve the transfer of somewhat more pages.

As the cylinder capacity of a 3330 disc pack is approximately 57 pages, both types of index file may reside within a cylinder. In practice, they may be otherwise distributed on account of the virtual storage system. Seek times for indexing information should be roughly similar for both retrieval systems. However, both rotational delay and data transfer time for sequential scans would be less per index

record than per primary index record.

### 5.2.1. Termination level of search path for non-existent keys

Both methods recognise the non-existence of data for a key if an index record does not exist for the partition. The in-core high-level array of the primary index also flags the non-existence of 10% data. Both methods could terminate search when keys are obviously outside the range of MINX and MAXX inclusive. Bitmap indexing offers not only an additional level at which data could be flagged as non-existent but also alternative methods of defining relevance, either in terms of suppression status or as the union, intersection of difference of the subsets.

### 5.2.2. The retrieval process

Sequential scan involves the examination of all intermediate records until the key or a higher value key is found. Bitmap indexing involves the testing of all intermediate bits. The minimum test ratio, i.e. the ratio of bit tests to records scanned, is unity, occurring when MAXX - MINX + 1 = ND. Gaps in the availability of data suggest that bitmap indexing is likely to incur more CPU time when most of the data present are relevant.

### 5.2.3. The selection ratio and move time

Efficiency of search is usually measured by the selection ratio (Engles, 1972), which is conventionally defined as the number of bytes selected or relevant to the number of bytes examined. In the context of the current problem and circumstances, the selection ratio (Rs) can be re-expressed as :

$$Rs = \frac{s}{m}$$

where s is the number of relevant bytes, and m is the number of bytes passed or moved from MTS I/O buffers to buffers within the retrieval system (the time to effect this move will be referred to hereafter as move time).

For the additional cost of bit-indexing, the primary index always ensures that Rs is unity by "reading " only relevant data. Hence, the larger the selection ratio by sequential scan the smaller the payoff in indexing.

## 5.3. Geographic Factors Influencing Performance

The relative efficiency of sequential scanning and bitmap indexing depends upon the density of data available and the characteristics of the spatial entity to be retrieved (see section 3.1 above). The location, size and form of the sub-area and the order in which its spatial identifiers are presented can influence the relative efficiency of search.

### 5.3.1. Order of identifiers

Sorting the keys into the order present in the data file has the effect of increasing the selection ratio, Rs, by sequential scan. All records are accessed in a single serial scan of the disc, thereby ensuring that pages which contain multiple records need be retrieved only once. Waters (1975), Cardenas (1975) and Pezarro (1976) discuss the effects of sorting on disc seeks. Both the time for retrieval from disc and the move time are minimised. Thus, retrieval via the primary index is faster than a sequential scan for unsorted identifiers, which may occur with radial searches and path tracking or traverses.

### 5.3.2. Form of spatial entity

Retrieval by both methods is again most rapid when the given set of co-ordinate references relates to a compact contiguous area, elongated in a latitudinal, rather than a longitudinal, direction. However, the primary index is especially useful for the speedy retrieval of data for dispersed locations. These access non-adjacent Y partitions, within which several records may separate those of interest, resulting in low selection ratios by sequential scan.

### 5.3.3. Size of search entity

In general, the smaller the size of the search entity (reckoned in number of keys presented), the greater the benefits of indexing.

### 5.3.4. Location of search area

The location of the sub-area of interest determines the data potential or the potential value of m (see 5.2.3 above). In sparsely populated areas, partitions based on Y values are likely to be shorter, owing to the small number and suppression status (determining the length) of records, while the test ratio is large. While these features favour a sequential scan, the proportion of keys for which data exist is also likely to be small. Conversely both potential m and test ratios are likely to be high in densely populated regions with a continuous distribution of people and/or households. These conditions favour bitmap indexing except when the search entity is so large and compact that, owing to the high potential for existing keys, s approaches m, producing high selection ratios by sequential scan. Thus it appears that bitmap indexing may be useful in the south of Britain, while its value is dubious in northern Scotland.

Furthermore, the search for data in the middle regions of a long partition (allowing for the backward processing capabilities under MTS) is likely to produce lower selection ratios by sequential scan than the search for the same quantity of data located at the front - or back-end of the same partition. Thus sequential scans may prove sufficient for data relating to the western and eastern coastal areas of Britain.

## 6. PERFORMANCE STATISTICS

The overheads of bitmap storage and indexing seem justified when both test ratios and selection ratios by sequential scan are low. It is most valuable when the intersection, union or difference of unsuppressed data for more than one record type is required for a small number of unsorted, dispersed one-kilometre squares. It may prove wasteful when data of one kind only are sought for a large number of pre-sorted adjacent one-kilometre squares in a relatively densely populated part of Britain. The possibilities of a mixed-mode retrieval system were contemplated and performance statistics were collected to identify the nature of the relationships between both systems and geographic location. The statistics were collected under conditions which were likely to produce comparable performances

by both systems. The search in each case was for the 100% population data only of a 100-kilometre block, i.e. for 10,000 keys. Thus a fifteenth or less of the data file was to be retrieved, and the records retrieved would be found adjacent to each other in a maximum of 100 separate logical blocks. The keys were pre-sorted into the optimal order. The run was repeated for each method and 100 km. block to evaluate the reliability of the retrieval times The CPU and elapsed times are given in Appendix 1. On the whole, the CPU times are less variable than the elapsed times, but even these are only reliable to the second As there are marked differences in total response and elapsed times and several instances of changes in the relative performance of both methods, inferences can only be tentative.

The indexing of bitmaps was expected to take more CPU time than sequential scans, as the minimum test ratio is unity. However, this in general seems to be apparent only towards the start of a data partition, where high selection ratios are bound to occur. South of the grid line 5000 metres, the high density of data produces lower test ratios. Concurrently, the selection ratios by sequential scan fall off progressively towards the end of the partition, where bitmap indexing becomes more profitable. Values for CPU and response time decrease at the end of the partition because several keys are outside the areas, i.e. MAXX, causing the search to terminate at an early stage. However, as the magnitude of difference is in general less than two seconds, the savings do not justify a mixed-mode retireval system, especially since the bulk of the time (about 5.9 seconds, which is the average of 16 runs) was used to read the 10,000 keys and perform the necessary accounting. Thus, under the worst conditions (for 100km square 5000 1000) less than nine seconds of CPU time is required to retrieve 8,541 records.

Figures for elapsed time are less reliable as they are dependent on concurrent activities in the system. The marked difference in response time between two identical jobs and reversals in the relative performance of both indexing methods (see Appendix 1) give some idea of the degree of fluctuation in elapsed times. The response rate per retrieved record is high when several keys are non-existent. However, in 42 out of the 54 100-km blocks, the response rate is less than one

tenth of a second per existing record. The observed response rates are adequate if the data for the sub-areas are infrequently accessed via the locational index, especially if data are required for smaller sub-areas. The data for County Durham, consisting of 2,105 records, can be retrieved in under three seconds via the locational index.

Speed of retrieval via the locational index can be improved by replacing the FORTRAN routines for processing the bitmaps with ASSEMBLER code, and by including additional pointers to the middle of long partitions. However, the bulk of the CPU time in 46 out of 54 100-km. blocks was spent on reading the 10,000 formatted keys, many of which were redundant, and on performing the necessary accounting. Moreover, the storage of 10,000 keys in 215 FORTRAN format requires 100,000 bytes, or approximately 25 pages of file space.

When data for a specified sub-area are to be retrieved repeatedly, considerable savings can be effected by constructing a compact and quick index with one pass through the locational index. This replaces the need for the list of X and Y references on subsequent runs. The 2,105 records for County Durham can then be retrieved in just over one second CPU time. It takes 26.8 seconds CPU time and about 176 seconds elapsed time to retrieve data for 30,000 primary X and Y co-ordinates in the three 100-km blocks 2000 3000, 3000 3000, 3000 4000 (Data are provided for only 17,110 of the 30,000 squares). On average (of three runs) it takes 7.4 seconds CPU time and 35 seconds elapsed time to retrieve the existing data for 17,110 records via a compact index. The overheads for constructing the compact index were about 28 seconds CPU and 83 seconds elapsed time. The latter requires 1,604 bytes (less than one page) for pointers and other associated indexing information and eliminates the need for the original keys and the locational index if data for the same area are repeatedly required. The strategy and design of the compact index and associated retrieval procedures will be discussed in a forthcoming paper.

## 7. CONCLUSION

The locational index is a low-level primary index. Both the index itself and the routines for manipulating it can be used directly by knowledgeable programmers. Most users of data, however, may only want a subset of data to be extracted and pre-processed for input to other existing packages for data analysis and display. The suite of routines for converting the users' compact description of subareas into the primary keys will be discussed in a forthcoming paper. A user may wish to store a small amount of only the relevant data in his own file space, but he may not have the resources to duplicate a large amount of data. When data for the same sub-area are repeatedly required, the locational index can be used to construct a compact area-index. The features of the compact area-index will be discussed in a separate paper.

The storage of the 1971 census data for Great Britain involved a complete change in the form and order of the data (Visvalingam and Perry, 1976). It was essential to check and double-check that no errors had been introduced during the CRU processing. The final checks involved a sequential scan through all the OPCS tapes. The locational index was indispensable for the purpose of comparing the content of every single OPCS record with that of the corresponding compacted CRU record.

The locational index was also very convenient for exercises in aggregation and for extracting data for pilot studies on small areas. It is repeatedly used for extracting small amounts of data for student classes. It also enables users to ascertain the availability of unsuppressed census data within sub-areas of interest to them.

The design of the locational index and the choice of retrieval strategies were determined to a great extent by the computing facilities available, especially the MTS file handling system, and the types of processing which were required. The indexing structure and its characteristics, described in Section 3.2. were tailored to specific limiting conditions; for example the lack of update problems, the availability of a disc pack for data storage, and the characteristics of the data (see Section 2).

The implementation of the structure was discussed in Section 4. The data structures used are by no means machine - or system-dependent, as they can be mapped onto a simple sequential organisation. The specific storage lay-outs were chosen to maximise performance under NUMAC. Decisions based on implementation - orientated features were pointed out.

Initially mixed-mode (sequential scanning and bitmap indexing) procedures for retrieval were envisaged. Observed performance statistics (Section 6) indicated that bitmap indexing on its own was adequate. The speed of retrieval would be further improved by replacing the FORTRAN code for bit processing by theASSEMBLER code and by using an optimising compiler such as FORTRANH rather than FORTRANG. The author is of the opinion that further efforts towards improving the design of the primary index is likely to yield minimal rewards. Greater benefits could be derived instead from procedures for restructuring search keys.

ACKNOWLEDGEMENTS

The Author is indebted to Mr. R.E. Vine of NUMAC for helpful information on the MTS file system and for correcting an earlier draft of the paper. She is also grateful to Mr. R. Sheehan of the Durham Computer Unit for his helpful comments on the paper and to Mrs. J. Dresser who typed the manuscript. The author is solely responsible for any remaining errors.

REFERENCES

BAXTER, R.S. (1976) Computer and Statistical Techniques for Planners, Methuen & Co. Ltd., London, 163-179.

CARDENOS, A.F. (1975) "Analysis and Performance of Inverted Data Base Structures", CACM, Vol. 18, No. 5, 253 - 263.

ENGLES, R.W. (1972) "A Tutorial on Data-Base Organisation", Ann. Review of Automatic Programming, Vol.7 No.2, 1-64

PEZARRO, M.T. (1976) "A note on estimating bit ratios for direct-access storage devices", Computer J., Vol.19, No.3, 271-272.

RHIND, D.W. (1975) "The State of Art in Geographic Data Processing a U.K. View", Proc. of the IBM UK Sc. Centre Seminar on Geographic Data Processing, Peterlee, Co. Durham, (ed. B.K. Aldred), 1-35

RHIND, D.W., EVANS, I.S. and DEWDNEY, J.C. (1977) "The derivation of new variables from population census data", Working Paper No.9, Census Research Unit, Department of Geography, University of Durham

VISVALINGAM, M. and PERRY, B.J. (1976) "Storage of the grid-square based 1971 G.B. Census data : checking procedures, Working Paper No.7, Census Research Unit, Department of Geography, University of Durham

WAGNER, R.E. (1973) "Indexing design considerations", IBM Syst. J., No.4, 851-367

WATERS, S.J. (1977) "Estimating magnetic disc seeks", Computer J., Vol. 18, No.1, 12-17.

## APPENDIX : PERFORMANCE STATISTICS

### KEY

#### Columns

1. Eastings of the 100-km square
2. Northings of the 100-km square
3. Number of records
4. CPU time, partitioned sequential scans
5. CPU time, bitmap indexing
6. Better method (see below)
7. Elapsed time, partitioned sequential scans
8. Elapsed time, bitmap indexing
9. Better method (see below)
10. Retrieval time per record found, partitioned sequential scans
11. Retrieval time per record found, bitmap indexing

#### Symbols (columns 6 and 9)

| | |
|---|---|
| S | partitioned sequential scans |
| B | bitmap indexing |
| = | both methods take approximately the same time |
| * | reversal in relative performance |
| + | marked difference in response time between runs |

APPENDIX (sheet 1)

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|
| 4000 | 12000 | 62 | 6.126 | 6.183 | = | 42.373 | 43.213 | S * | 0.683 | 0.697 |
| | | | 6.020 | 5.964 | = | 32.623 | 31.263 | B | 0.526 | 0.504 |
| 3000 | 11000 | 8 | 6.099 | 6.221 | = | 48.753 | 43.756 | B + | 6.094 | 5.469 |
| | | | 6.053 | 6.037 | = | 27.636 | 25.860 | B | 3.454 | 3.232 |
| 4000 | 11000 | 583 | 6.326 | 6.404 | = | 50.190 | 46.373 | B + | 0.086 | 0.080 |
| | | | 6.183 | 6.230 | = | 29.616 | 26.856 | B | 0.051 | 0.046 |
| 2000 | 10000 | 1 | 6.060 | 6.161 | = | 40.786 | 42.493 | S + | 40.786 | 42.493 |
| | | | 6.062 | 6.039 | = | 42.996 | 42.266 | B | 42.996 | 42.266 |
| 3000 | 10000 | 701 | 6.306 | 6.391 | = | 45.970 | 51.900 | S | 0.066 | 0.074 |
| | | | 6.232 | 6.281 | = | 36.460 | 39.206 | S | 0.052 | 0.056 |
| 4000 | 10000 | 5 | 6.121 | 6.213 | = | 43.886 | 29.476 | B | 8.777 | 5.895 |
| | | | 6.037 | 6.039 | = | 37.136 | 30.863 | B | 7.427 | 6.173 |
| 0 | 9000 | 9 | 6.024 | 6.186 | = | 29.396 | 29.770 | S | 3.266 | 3.308 |
| | | | 5.998 | 6.057 | = | 31.370 | 35.663 | S | 3.486 | 3.963 |
| 1000 | 9000 | 373 | 6.308 | 6.459 | = | 29.980 | 39.240 | S | 0.080 | 0.105 |
| | | | 6.287 | 6.315 | = | 38.463 | 40.763 | S | 0.103 | 0.109 |
| 2000 | 9000 | 603 | 6.505 | 6.598 | = | 31.686 | 30.440 | B * | 0.053 | 0.050 |
| | | | 6.423 | 6.507 | = | 37.426 | 40.133 | S | 0.062 | 0.067 |
| 3000 | 9000 | 769 | 6.459 | 6.628 | = | 33.306 | 34.250 | S * | 0.043 | 0.045 |
| | | | 6.324 | 6.408 | = | 36.166 | 33.290 | B | 0.047 | 0.043 |
| 0 | 8000 | 314 | 6.379 | 6.533 | = | 29.783 | 30.840 | S | 0.095 | 0.098 |
| | | | 6.167 | 6.260 | = | 26.470 | 34.023 | S | 0.084 | 0.108 |
| 1000 | 8000 | 842 | 6.754 | 6.917 | = | 31.613 | 32.786 | S * | 0.038 | 0.039 |
| | | | 6.633 | 6.654 | = | 38.746 | 37.716 | B | 0.046 | 0.045 |
| 2000 | 8000 | 2028 | 7.099 | 7.320 | S | 25.460 | 26.283 | S | 0.013 | 0.013 |
| | | | 7.087 | 7.273 | = | 32.483 | 33.586 | S | 0.016 | 0.017 |
| 3000 | 8000 | 4380 | 7.996 | 8.130 | = | 32.853 | 34.153 | S * | 0.008 | 0.008 |
| | | | 7.879 | 8.003 | = | 41.513 | 34.390 | B | 0.009 | 0.008 |
| 4000 | 8000 | 344 | 7.146 | 7.145 | = | 29.733 | 34.030 | S * | 0.086 | 0.099 |
| | | | 7.130 | 7.040 | = | 36.380 | 34.350 | B | 0.106 | 0.100 |
| 0 | 7000 | 61 | 6.076 | 6.247 | = | 28.646 | 29.236 | S | 0.470 | 0.479 |
| | | | 6.010 | 6.101 | = | 26.930 | 36.386 | S | 0.441 | 0.596 |
| 1000 | 7000 | 899 | 6.626 | 6.759 | = | 30.803 | 30.446 | B * | 0.034 | 0.034 |
| | | | 6.551 | 6.661 | = | 28.770 | 30.706 | S | 0.032 | 0.034 |
| 2000 | 7000 | 1698 | 6.967 | 7.103 | = | 32.786 | 32.063 | B * | 0.019 | 0.019 |
| | | | 7.057 | 7.146 | = | 29.503 | 34.696 | S | 0.017 | 0.020 |

APPENDIX (contd. sheet 2)

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|
| 3000 | 7000 | 3949 | 7.766 | 7.909 | = | 25.903 | 51.993 | S * | 0.007 | 0.013 |
| | | | 7.875 | 8.029 | = | 38.643 | 34.426 | B | 0.010 | 0.009 |
| 4000 | 7000 | 0 | 6.035 | 6.163 | = | 30.113 | 36.040 | S + | 30.113 | 36.040 |
| | | | 5.963 | 6.119 | = | 13.213 | 13.870 | S | 13.213 | 13.870 |
| 1000 | 6000 | 900 | 6.653 | 6.723 | = | 38.876 | 41.530 | S | 0.043 | 0.046 |
| | | | 6.653 | 6.753 | = | 34.766 | 40.410 | S | 0.039 | 0.045 |
| 2000 | 6000 | 5102 | 8.398 | 8.574 | = | 49.943 | 39.813 | B | 0.010 | 0.008 |
| | | | 8.632 | 8.759 | = | 46.596 | 41.506 | B | 0.009 | 0.008 |
| 3000 | 6000 | 4084 | 8.797 | 8.711 | = | 58.123 | 51.026 | B | 0.014 | 0.012 |
| | | | 8.904 | 8.818 | = | 60.323 | 55.956 | B | 0.015 | 0.014 |
| 4000 | 6000 | 654 | 7.240 | 7.222 | = | 57.620 | 59.313 | S + | 0.088 | 0.091 |
| | | | 7.323 | 7.249 | = | 32.553 | 36.220 | S | 0.050 | 0.055 |
| 1000 | 5000 | 47 | 6.078 | 6.254 | = | 35.583 | 36.160 | S * | 0.757 | 0.769 |
| | | | 6.128 | 6.211 | = | 42.786 | 41.743 | B | 0.910 | 0.888 |
| 2000 | 5000 | 2372 | 7.006 | 7.149 | = | 40.486 | 41.133 | S | 0.017 | 0.017 |
| | | | 7.147 | 7.253 | = | 44.276 | 47.206 | S | 0.019 | 0.020 |
| 3000 | 5000 | 5001 | 8.375 | 8.500 | = | 49.923 | 50.056 | = | 0.010 | 0.010 |
| | | | 8.586 | 8.697 | = | 35.966 | 38.480 | S | 0.007 | 0.008 |
| 4000 | 5000 | 4165 | 8.841 | 8.780 | = | 58.500 | 57.416 | B + | 0.014 | 0.014 |
| | | | 8.902 | 8.840 | = | 31.013 | 30.503 | B | 0.007 | 0.007 |
| 3000 | 4000 | 5225 | 8.565 | 8.757 | = | 32.650 | 33.936 | S * | 0.006 | 0.006 |
| | | | 8.416 | 8.597 | = | 49.816 | 49.276 | B | 0.010 | 0.009 |
| 4000 | 4000 | 7567 | 10.824 | 10.811 | = | 63.093 | 58.056 | B | 0.008 | 0.008 |
| | | | 10.724 | 10.731 | = | 75.686 | 74.936 | B | 0.010 | 0.010 |
| 5000 | 4000 | 1607 | 9.597 | 9.093 | B | 66.163 | 64.056 | B | 0.041 | 0.040 |
| | | | 9.619 | 9.011 | B | 72.546 | 70.280 | B | 0.045 | 0.044 |
| 2000 | 3000 | 3239 | 7.440 | 7.598 | = | 44.346 | 44.156 | = + | 0.014 | 0.014 |
| | | | 7.366 | 7.551 | = | 48.790 | 69.390 | S | 0.015 | 0.021 |
| 3000 | 3000 | 8646 | 10.575 | 10.629 | = | 81.960 | 85.996 | S | 0.009 | 0.010 |
| | | | 10.455 | 10.594 | = | 77.180 | 83.286 | S | 0.009 | 0.010 |
| 4000 | 3000 | 8221 | 12.775 | 12.075 | = | 100.240 | 76.130 | B * | 0.012 | 0.009 |
| | | | 12.964 | 12.611 | B | 89.670 | 106.853 | S | 0.011 | 0.013 |
| 5000 | 3000 | 5248 | 13.068 | 12.675 | B | 88.650 | 94.596 | S * | 0.017 | 0.018 |
| | | | 13.122 | 12.215 | B | 73.666 | 62.120 | B | 0.014 | 0.012 |
| 6000 | 3000 | 1734 | 9.545 | 9.059 | B | 72.090 | 81.450 | S | 0.042 | 0.047 |
| | | | 9.580 | 9.115 | B | 55.613 | 54.333 | B | 0.032 | 0.031 |

APPENDIX (contd. sheet 3)

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1000 | 2000 | 652 | 6.405 | 6.591 | = | 45.330 | 45.386 | = | 0.070 | 0.070 |
| | | | 6.360 | 6.495 | = | 39.200 | 40.736 | S | 0.060 | 0.062 |
| 2000 | 2000 | 5269 | 8.176 | 8.347 | = | 53.523 | 61.036 | S | 0.010 | 0.012 |
| | | | 7.994 | 8.106 | = | 53.253 | 54.306 | S | 0.010 | 0.010 |
| 3000 | 2000 | 8581 | 10.724 | 10.775 | = | 75.686 | 74.936 | B * | 0.009 | 0.009 |
| | | | 10.376 | 10.464 | = | 54.933 | 62.200 | S | 0.006 | 0.007 |
| 4000 | 2000 | 8074 | 13.165 | 12.532 | B | 92.580 | 84.160 | B * | 0.011 | 0.010 |
| | | | 12.896 | 12.453 | B | 81.503 | 87.910 | S | 0.010 | 0.011 |
| 5000 | 2000 | 7952 | 15.665 | 14.584 | B | 96.286 | 70.536 | B | 0.012 | 0.009 |
| | | | 15.100 | 14.326 | B | 98.816 | 98.326 | B | 0.012 | 0.012 |
| 6000 | 2000 | 3291 | 13.665 | 12.521 | B | 81.013 | 78.643 | B + | 0.025 | 0.024 |
| | | | 13.733 | 12.430 | B | 116.340 | 109.010 | B | 0.035 | 0.033 |
| 1000 | 1000 | 42 | 6.132 | 6.273 | = | 30.466 | 29.440 | B | 0.725 | 0.701 |
| | | | 6.024 | 6.165 | = | 29.280 | 29.306 | = | 0.697 | 0.698 |
| 2000 | 1000 | 3706 | 7.518 | 7.688 | = | 34.900 | 35.940 | S | 0.009 | 0.010 |
| | | | 7.484 | 7.704 | S | 35.606 | 36.286 | S | 0.010 | 0.010 |
| 3000 | 1000 | 7669 | 10.150 | 10.194 | = | 48.876 | 48.060 | B | 0.006 | 0.006 |
| | | | 10.402 | 10.463 | = | 49.446 | 49.550 | = | 0.006 | 0.006 |
| 4000 | 1000 | 7934 | 12.671 | 12.375 | B | 64.616 | 75.090 | S | 0.008 | 0.009 |
| | | | 12.994 | 12.698 | B | 62.633 | 65.290 | S | 0.008 | 0.008 |
| 5000 | 1000 | 8541 | 15.141 | 14.433 | B | 103.503 | 99.780 | B | 0.012 | 0.012 |
| | | | 14.995 | 14.654 | B | 115.123 | 98.630 | B | 0.013 | 0.012 |
| 6000 | 1000 | 1260 | 12.355 | 11.488 | B | 96.486 | 90.216 | B * | 0.077 | 0.072 |
| | | | 12.391 | 11.470 | B | 75.953 | 103.950 | S | 0.060 | 0.082 |
| 0 | 0 | 20 | 6.008 | 6.144 | = | 46.010 | 48.303 | S | 2.300 | 2.415 |
| | | | 6.188 | 6.251 | = | 36.730 | 40.916 | S | 1.836 | 2.046 |
| 1000 | 0 | 1682 | 6.719 | 6.652 | = | 51.426 | 53.703 | S | 0.031 | 0.032 |
| | | | 7.077 | 7.207 | = | 46.593 | 46.706 | = | 0.028 | 0.028 |
| 2000 | 0 | 4148 | 7.821 | 7.902 | = | 58.576 | 50.256 | B * | 0.014 | 0.012 |
| | | | 8.329 | 8.455 | = | 47.916 | 52.793 | S | 0.012 | 0.013 |
| 3000 | 0 | 1295 | 7.178 | 7.172 | = | 43.486 | 42.016 | B * | 0.034 | 0.032 |
| | | | 7.669 | 7.617 | = | 38.400 | 44.270 | S | 0.030 | 0.034 |
| 4000 | 0 | 839 | 7.082 | 7.178 | = | 45.940 | 44.290 | B | 0.055 | 0.053 |
| | | | 7.475 | 7.610 | = | 44.460 | 40.626 | B | 0.053 | 0.048 |
| 5000 | 0 | 44 | 6.294 | 6.361 | = | 39.660 | 40.180 | S * | 0.901 | 0.913 |
| | | | 6.577 | 6.673 | = | 31.133 | 27.450 | B | 0.708 | 0.624 |

STOP 0
EXECUTION TERMINATED