APPLICATIONS OF LATTICE THEORY TO AUTOMATED CODING AND DECODING

Lawrence H. Cox
U.S. Bureau of the Census

INTRODUCTION

The potential of a data base lies beyond its being a repository of coded facts or observations. The data items give form and structure to the information content of the data base, which includes the totality of all facts and conclusions derivable from the interrelationships between data items relative to the data base as a whole. In many applications, this information can be meaningfully interpreted in terms of the set-theoretic hierarchy generated by the data set. In these cases, mechanisms to construct and analyze this hierarchy are necessary to the efficient manipulation and utilization of the data. This paper describes techniques and a computer system for examining the hierarchical relationships which exist between component subsets of the data set and for organizing, managing and accessing the data base in a manner consistent with this hierarchy. The system employs the advantages of a random-access computing environment.

The impetus for the research and the initial application of the resulting computer system is in the area of geocoding and statistics dissemination. A geographic region, such as the United States, has been partitioned in several ways along different lines: geographic (states, counties), economic (standard metropolitan statistical areas, major retail centers), political (congressional districts) and demographic (urbanized and rural areas, census tracts). Each basic region (set) in these partitions is assigned a unique code. Data keyed to these geocodes and perhaps to other (subject matter) codes are collected and statistics are tabulated and published on the basis of the coding hierarchy thus formed. The problem which then arises is that of identifying all statistical inferences derivable from these publications, thereby capturing a major portion of the total information content of the data base. This amounts to constructing the lattice representation of the Boolean algebra generated by the codes. This lattice is of particular importance in editing the tabulations to improve the quality and statistical confidentiality of the final publications.

THE DATA STRUCTURE

$\underline{X}$ denotes a finite set and $X(i) = \left\{X(i,j): \bigcup_j X(i,j) = X,\ X(i,j) \cap X(i,l) = \emptyset \text{ for } j \neq l\right\}$ partitions of $\underline{X}$, $i = 1,\dots\dots, m$. Each $x \in X$ is assigned a unique code $(j(1),\dots\dots,j(m))$, indicating that $x \in X(i,j(i))$ for $i = 1,\dots\dots,m$. Henceforth, $\underline{x}$ will be synonymous with its corresponding code. We identify all containment and equality relationships $X(i,j)\subseteq X(k,l)$, $X(i,j)=X(k,l)$, and, more generally, those partitions $X(i)$ which are refinements of other partitions $X(k)$. The goal is a data structure and retrieval system which reflects the characteristics and set-wise interrelations of the data items and allows for efficient manipulation and management of the data base.

The input file consists of data records, each keyed to a unique $x\in X$. A file image of the reference set $\underline{X}$ is created, each record of which has the form $(i(1),\ldots.,i(m),N)$, where $(i(1),\ldots.,i(m))$ is the code corresponding to $\underline{x}$ and $\underline{N}$ is the unique record number identifier of the record $\underline{x}$, determined by a fixed enumeration of the elements of $\underline{X}$.

To each basic set $X(i,j)$ are associated $\underline{m}$ lists $L(i,j;s)$. $L(i,j;i)$ is an ordered sequence of record numbers of records corresponding to elements $x\in X(i,j)$. For $s\neq i$, $L(i,j;s)$ is an ordered list of all $X(s,l)$ for which $X(i,j)\cap X(s,l)\neq\emptyset$. These lists are formed by sorting the file $\underline{X}$ $\underline{m}$ times, once on each field/record number. Subsequent to the first sort, the final record numbers $\underline{N}$ are assigned. These lists, as modified and manipulated by the system described below, are sufficient to perform all set-wise analyses. No further sorting of the file is necessary to query or update the data base.

Containment and equality between basic sets in different partitions are easily identified: $X(i,j)\subseteq X(k,l)$ if, and only if, the only entry in $L(i,j;k)$ is $X(k,l)$. Pairs of opposite inclusions are equivalent to equality.

Unions and intersections are easily computed via list merging and matching techniques. As the lists are increasing sequences of integers, these operations are quickly and efficiently performed in a minimum of core. To intersect the sets $X(i,j)$ and $X(k,l)$, the list $L(i,j;k)$ is referenced to determine whether the intersection is void or not. If not, two elements from $L(i,j;i)$ and one element from $L(k,l;k)$ are consecutively flashed into core. If the element from the second list equals either of the elements from the first list, this element is removed from both lists and is added to the intersection list and the next element from both lists are brought into core. If the element from the second list is less than the larger element from the first list and unequal to the smaller, it is discarded and the next element from the second list is brought into core. If the element from the second list is larger than the larger element from the first list, the smaller element from the first list is discarded and the next element from the first list is brought into core. This process is repeated until the larger element from the first list is greater than or equal to the element from the second list. The comparisons above are then made and the procedure proceeds recursively until either list is exhausted. Set-wise unions may be computed in an analogous manner.

All sets constructible from the basic sets are expressible as disjoint unions of the sets $Y(v) = \bigcap_{i=1}^{m} X(i,j(i))$, $v = (j(1)),\ldots.,j(m))$. The sets $Y(v)$ are the atomic elements in the lattice generated by the partial ordering induced by the codes. The $Y(v)$ are computed and to each is associated an increasing sequence of the record numbers of the records $x\in Y(v)$. The lists $L(i,j;i)$ are then replaced by ordered lists of those $\underline{v}$ for which $Y(v)\subseteq X(i,j)$ (i.e. $Y(v)\cap X(i,j)\neq\emptyset$). The resulting data structure is sufficient to perform all set-theoretic constructions and comparisons relative to this hierarchy in a quick and computationally efficient manner.

The computation of intersections is performed as described above, the only difference being that the lists being merged are ordered lists of subsets rather than elements. This consolidation improves the speed and minimizes the storage requirements of the intersection process. Similarly, set-wise unions are computable in relatively few operations and the problem of alternative descriptions is rendered tractable. More precisely, let $U = \bigcup_{i \in I, j \in J} X(i,j)$ denote a union of basic partition sets over index sets I and J (e.g., a union of certain counties and places within a State). It is meaningful and frequently necessary to determine whether U is expressible in a different (perhaps more succinct) manner (e.g., as a standard metropolitan statistical area). Working only with element by element descriptions of the sets X(i,j), the lists L(i,j;s) for $s \neq i$, and the remaining sets in the universe, this problem can be solved only through an exhaustive sequence of list comparisons. The redefined L(i,j;i) greatly improve this situation and facilitate the computation. For each $p \notin I$, a list G(p) of those $\underline{q}$ appearing in field $\underline{p}$ of any L(i,j;i), $i \in I$, is formed. This amounts to grouping for each $p \in I$ those sets X(p,q) which intersect $\underline{U}$. The sets $U(p) = \bigcup_{z \in G(p)} z$ are all upper bounds of $\underline{U}$, each along a particular component direction in the hierarchy. The U(p) are of interest in and of themselves from a statistics dissemination standpoint. Moreover, any alternative description of $\underline{U}$ must be expressible in terms of the U(p). In particular, a "more succinct description" of $\underline{U}$ could generally be defined as U(p) for which U = U(p) and U(p) = X(p,q). These are easily identified.

This data structure can be maintained and updated routinely. The addition or deletion of records from the reference set $\underline{X}$ will alter the definition of the Y(v), but as all sets X(i,j) containing a given Y(v) are characterized by the condition v(i) = j, all necessary edits and alterations to the data structure may be identified and effected in a straightforward manner.

INITIAL IMPLEMENTATION AND APPLICATIONS

The reference set $\underline{X}$ is a universe of some 9,200 basic records which comprise the geographic reference file employed by the U.S. Bureau of the Census in tabulating statistics for the 1972 Census of Manufactures. This set represents a geocoding of the United States into 50 States and approximately 3,100 counties, 5,900 places and 264 standard metropolitan statistical areas. The programs are written in Fortran IV and the system will soon be run for testing in an interactive environment on a PDP-10 computer. The system will subsequently be implemented on a Univac 1110.

One of the responsibilities of a statistics gathering and disseminating body such as the U.S. Bureau of the Census is maintaining statistical confidentiality in protecting the identities and responses of individual respondents. If a candidate cell for publication is deemed a disclosure and is to be suppressed from publication, the remaining cells in the publication tables must be examined to determine to what extent they can be employed to compute estimates of the suppressed cell. If the finest such estimate computable is also deemed a breach of statistical confidentiality, additional cells must be suppressed until an acceptable equivocation of the value of this statistic is reached. These additional suppressed cells must then be protected in like fashion. The result is an extremely complex logical and computational problem fueled by the cascading effect of suppressions and estimates proliferating through the publication hierarchy. The only way to effectively manage and control this process and strike an acceptable balance between the conflicting goals of maintaining statistical confidentiality and maximizing the quantity and quality of the published information is through the construction and analysis of the set-theoretic lattice generated by the publication hierarchy. The lattice provides a natural mechanism for defining the order of processing and for driving the tabulation programs, while computing best estimates of suppressed cells to assure that confidentiality is maintained. This is an intended purpose of the system described in this article. Its other major function is as a standing geographic reference file, with obvious extensions and applications to any logical hierarchy generated by a set of codes.

ACKNOWLEDGEMENTS

Throughout the course of this research, the author has collaborated on a continuing basis with Patricia Griffin and Marvin White of the U.S. Bureau of the Census. Mrs. Griffin is responsible for programming the system, Mr. White provided expertise in the area of data processing in a random-access environment, and both individuals have made valuable contributions to the conceptual and computing aspects of the problem. The author acknowledges their contributions with gratitude.