A Computer System for Urban Studies

WREN McMAINS and AARON FLEISHER, Department of City and Regional Planning, Massachusetts Institute of Technology

•SOCIAL sciences are not generally thought of as laboratory sciences, and social scientists seem content to find their experiments wherever they can. Not being of their own making, such experiments are rarely pointed, and sometimes not even to the point. This disability is so common that it is considered natural and inevitable. It is neither. We do not mean to infer that social science laboratories can attain the elegance, facility and finality of the physical sciences. We do assert, however, that the gap can be closed enough to make the enterprise well worthwhile.

Our purposes will be better understood if we describe first the context in which the system is embedded. We are working toward a laboratory of urban studies, and we intend to equip our laboratory with these facilities:

1. A data library organized for both ease and efficiency of access, search, retrieval, editing and expansion;
2. A library of mathematical programs by which these data can be analyzed and manipulated;
3. A set of graphical procedures by which results can be presented;
4. A data structure by which to describe urban problems and designs;
5. A set of orders and procedures which would help in the design and operation of experiments and simulations; and
6. A repertoire of programs and a facility for programming to help in the analysis, design and evaluation of plans and decisions.

All the facilities will be programmed for a modern digital computer operating as a time-shared machine. A computer operating in this mode allows real-time communication between the user and the system without using more computer time than that required for computation. While the computer is waiting for a request from one user, it can service other users at remote locations. The user can start and direct all operations from his control console. He will be able to examine the data before using them, to look at intermediate steps in the analysis and processing, to edit, modify, reconsider, or resume the process. It will be possible to use almost all parts of the system in a batch processing mode; however, this limits the intercourse between computer and user. This mode of operation is useful for production runs that become routine when the experiment is completely designed. In either mode of operation, these facilities can be used separately or sequentially.

SYSTEM SPECIFICATIONS

From the outset it was clear that the system must meet two prime specifications. Most of its users will know little mathematics and less programming. Moreover, they will be unaccustomed to operating in a context where any mistake, however small, is catastrophic. They are not likely to appreciate that punctuation, spelling, form and order must be precise. They are also likely to be easily discouraged. Therefore, the system must guide its user along his way. It ought not to stop sullenly when it encounters mistakes. The system must be capable of calculating the costs and other ramifications of what it is the user asks for so that the user can decide if he really

Paper sponsored by Committee on Land Use Evaluation and presented at the 46th Annual Meeting.

wants what he is about to get. Yet the system must also be capable of responding to the subtleties of the capable programmer.

The second is a specification on the library. We intend a data library whose acquisition policy is the same as that of a university library. The purpose is a complete rather than a critical collection. The library therefore will be large and its users heterogeneous. A computer-operated index and acquisitions system then becomes necessary.

Therefore, we have decided to accept the data in the form that they arrive. No structure is imposed a priori. That option is left to the user. At present he is able to order his data in multidimensional tables, which is a simple kind of list structure. In the future he will be able to avail himself of more elaborate list processes. The index, however, will be highly structured, which is to say that structure will be imposed on the names and characteristics of the data and not on the data themselves. The index will act somewhat as a card catalogue except that it will be capable of far finer discriminations and be more easily modified. The retrieval system is intended to be able to follow all the ramifications contained in the index.

Acquisitions and Indexing in the Data Library

The library must be able to acquire data in any machine-readable form and, if necessary, to edit and correct them. Its capabilities will rest largely in the index provided for search, and the procedures for retrieval and display.

All data files can be accepted and entered as they arrive. Only the mode in which a file is written and the location and kinds of information need be described in order to make the file immediately readable. If it is considered desirable, the file can be rewritten into a form which can be read faster.

A description of the contents of each data file will be entered into the index. The index must list the names by which the variables are commonly known, the geographic places to which they refer, their dates, sources and statistical characteristics. Each category will be listed separately and all of them cross-referenced. In the process of assimilating a set of data into the library and correcting, editing or rewriting them, the index will simultaneously be modified. All this machinery will operate as part of the procedure of acquisition.

Searching the index will proceed by way of a systematic interrogation of the computer. The searcher, for example, may start by asking, "What do you have on income by industry?" How the question is worded will not be critical. The computer may answer by asking, "Where?" The searcher might say "Boston." Then the computer would produce the names and a brief description of the sources containing such data. The searcher would pick what he wants, or ask for further details, or go back to a higher level in the hierarchically structured index.

We are not certain how best to order these questions in logical nets. Rather than work toward some figure of merit given a priori, we propose a subsystem that would observe the use of the index and make improvements a posteriori. Then it would be simple to keep count of the categories searched. This information is important where the library is used routinely. An old fashioned card catalogue of the more frequently requested categories could be a more efficient way to display them. Thus we are not committed to the computer for all purposes.

It would be more complicated, but not very difficult, to record the paths taken through the index. Efficiency would be increased by easing those paths more frequently taken. They could, for example, be replaced by direct links. The index may also become more useful if, to each variable, were attached the names of all the other variables it has been coupled with in the making of tables. Easing paths through the index and augmenting the list of cross referents can be changes that the system itself would work, in which case one could say that the index "learns" and incorporates its efficiences.

The Retrieval Procedures

For our purposes, retrieval requires programs that call information from one of several very large files of surveys, such as the individual responses from a transportation study or the one in a thousand U.S. Bureau of the Census tape. Because the files are so large, speed and efficiency are extremely important, and for this reason it may be necessary to rewrite some files into a form which can be read faster.

We have decided on table making as a basic device for data retrieval. To obtain a table one needs only to list the variables as they are named by the index and order "tabulate." The system is presently capable of producing tables of 35 dimensions. That seems large enough. New variables that are transformations of the old ones can be created and used when making tables. The tables can be printed, displayed visually, or stored for future reference and manipulation.

Tables will not be the only mode of presentation. Curves and histograms will also be possible choices. And since urban problems frequently turn upon patterns of location, we can also display data values on geographical maps.

The Manipulative Procedures

We shall have on immediate call a repertoire of programs to operate on the data files that the user assembles in the form of tables. These are the kinds of manipulations we have in mind:

1. Arithmetic;
2. Function generators: logarithmic, exponential, polynomial, trigonometric, etc;
3. Matrix algebra;
4. Statistical analysis: moments, such as means and variances, tests of hypotheses, limits of confidence, and measures of significance;
5. Regression: simple, multiple and simultaneous; principal component, factor and discriminant analysis;
6. Curve fitting;
7. Graphics: data maps, curve tracing, histograms;
8. Mathematical optimation techniques (including linear programming); and
9. Et cetera.

We mean the "et cetera" literally. The system shall be devised for the easy integration of any program, prepackaged or specially tailored, or for the immediate addition of a datum without necessarily having to go through the library routines.

The results of such additions and manipulations can be retained in the private file of the user who makes them, to be called on as he wishes, or they can become a part of the public files of the general library by routing them through the acquisitions and indexing subsystem. The library in all its parts, therefore, will grow by its use.

Simulations and Experiments

A laboratory cannot be made from a set of experiments prepared a priori. It must be able to accommodate easily at least a wide range of purposes. We meet this requirement in two ways. It will be possible to write computer programs within the system, and possible to fill prescribed data structures from the files in the library.

Simulation is a kind of experiment. We have singled it out for separate mention because it is peculiarly a product of the computer's capabilities. We will include the basic structure for many simulations and the means for linking new simulations to the existing system.

The experiment is the user's choice, but we anticipate that the following kinds may be of general interest.

1. The computational machinery for making analyses of urban and regional economies, such as sector, economic base, and input-output.
2. The analysis, allocation and assignment algorithms in transportation networks. In particular we shall further explore aggregative behavior as a function of individual utility.
3. The inverse power and exponential allocation rules for the distribution of travel and land-use assignments.
4. Optimal location and trip distribution as obtained from linear programming formulations.

Analysis, Design and Evaluation of Decisions

At present, most of the time we have allocated to this matter is occupied with the analysis and evaluation of the visual qualities of physical designs. For this purpose we are learning to program the computer to act as a hemi-demi-semi literate draftsman. To do this in a way that is most convenient for a designer, both the input and the output of the machine must be in the form of drawings. That is the technical problem. The analytical problem requires a decision on what kind of drawings to make. We have tentatively settled on sequences of perspective taken along a specified path through the design. These perspective drawings will be the raw material for studies in perception and visual quality.

We have also begun to work on the problems in the processes of design itself. The specific problem we have posed is to devise methods by which to construct sets of rules (i.e., algorithms) that would replicate the product of design. An analysis of the processes of design would come out of the analysis of those sets of rules that are successful replicators. Concentrating on replication is an indirect route to examining the process of design, but it is one that has the advantage of relatively clear verification. It has been chosen for this reason.

THE PRESENT STATUS OF THE SYSTEM

Let us now consider in more detail the system as it exists today. First, we shall give examples of some of the existing requests and their uses. Suppose a person wanted to find information on mobility in the Boston area—in particular a table showing length of residence vs the family income level. The commands would be these:

```
USE/FILE    BOSTON/HOUSEHOLD
TABULATE    MOBILITY    YEARS/AT/ADDRESS.VS.INCOME/LEVEL
EXECUTE
R/P/PRINT   MOBILITY
C/P/PRINT   MOBILITY
```

The "USE/FILE" request tells what data file is to be used ("BOSTON/HOUSEHOLD"). The "TABULATE" statement orders a table where the rows are length of residence and the columns are different income levels, and calls this table "MOBILITY." The "EXECUTE" request says that we have specified all the tables required at this time, go ahead and make them. The last two requests say that we want the table printed twice, first as percentages of row totals and then as percentages of column totals.

If we are now interested in comparing these results for the region with a specific area in the region, we might write these commands:

```
USE/FILE    BOSTON/HOUSEHOLD
COMBINE     YEARS/AT/ADDRESS  0-1, 2-5, 6-10, 11-99
COMBINE     RESIDENCE/ZONE-INTO-DORCHESTER  61, 62, 63, 64, 65, 107
COMBINE     RESIDENCE/ZONE-INTO-NEW/RES  61-65 + 107, 1-60 + 66 - 106 + 108 - 626
TABULATE    MOBIL/1    DORCHESTER.VS.YEARS/AT/ADDRESS.VS.INCOME/LEVEL
TABULATE    MOBIL/2    NEW/RES .VS. YEARS/AT/ADDRESS .VS. INCOME/LEVEL
EXECUTE
PRINT       MOBIL/1
R/P/PRINT   MOBIL/2
PRINT       MOBIL/2
```

We are using the same data file as before, but this time we have decided to change (or combine) "YEARS/AT/ADDRESS" into a variable with the same name and four levels, or lengths of residence, instead of one hundred. We recombine the variable "RESIDENCE/ZONE" into two new variables. The first is called "DORCHESTER" and will have six levels, or zones in this case. The second, called "NEW/RES", will have only two levels—one containing all zones in Dorchester, the other containing all other zones in the region. We then ask for two 3-dimensional tables to be made. The first will be printed as six 2-dimensional tables with the four new levels of "YEARS/AT/ADDRESS" as rows and the income levels as columns. (There are six of these tables because there are six levels of "DORCHESTER.") The second table, which is really two 2-dimensional tables, will be printed twice, once giving percentages of row tables, and again giving absolute values.

Because it is an on-line system, the user will get almost immediate response in the form of advice, reprimands, and rewards. The system is able to inform the user of the consequences of his requests (amount of information that will be generated and the cost) before executing the requests. This will limit the amount of useless information generated by misunderstandings on the part of the user. The reprimands administered by the system are gentle and meant to show the user the error in his ways.

The critical time in a system of this type is the time required to operate on each logical record, not the time used in reading requests. The user speed is very slow relative to the time required for interpreting his requests. Some of this waiting time, therefore, is used to set up very efficient programs which will be executed as the data is read (remember that many tables can be made in one pass of the file). The marginal cost of the extra tables is very low because the system is input/output bound during this phase, unless a very large number of tables are requested. This is accomplished by double buffering, which allows the system to operate at read speed most of the time.

The following operations are performed as each logical record is read: (a) unpacking and placing the data in the variable list; (b) combining a variable; (c) arithmetic operations on variables; (d) user written subroutines; and (e) incrementing the tables. From these, programs are compiled to perform a and c as the requests are made; b and e are accomplished by pre-programmed routines which are supplied parameters and parameter lists indirectly via the requests; whereas d makes use of routines compiled by other compilers available for general use via the time-sharing system (e.g., MAD, FORTRAN, ALGOL, assembly language, etc.). We supply no general compilation features—this has been done very well by others—but we do supply several special purpose compilers indirectly.

Once tables have been made, they can be added, subtracted, multiplied and divided element by element by other tables. They can be restructured, or saved for future manipulations. They can be printed as absolute values or as percentages of totals.

Rows, columns, planes, etc. are labeled as they are printed—if labels are available—otherwise, numeric labels are used. Labels contained with the file are always available automatically. Labels for variables created by the user can be entered by the user before the table is printed, and will be used in every table in which this variable occurs.

The tables can be written on the disk for delayed printing, or saved on the disk in their internal form for later use. It is also possible for the user to save the entire system in its present state so he can return to it at another time.

The system can handle its own allocation of table storage space, or it can be assisted by the user, thereby taking advantage of his knowledge of future requests. An N-dimensional table where the i th dimension has n_i levels requires $\prod_{i=1}^{N} (n_i) + N + 1$ locations. The N + 1 locations contain pointers to names and labels, and dimensionality information. All of the tables are stored in core if room is available. If not, then room is made by shifting tables which are not required at the moment to secondary storage. Under user control, the user is informed of the overflow and he then has several options: (a) deleting tables for which he has no further use; (b) moving tables to permanent secondary storage (leaves tables in internal form, but saves them for use on another day); (c) moving tables to temporary secondary storage (same storage used in automatic mode); and (d) continuing in either manual or automatic mode.

Automatic rearrangement of storage is done only when necessary. Sometimes it will only involve primary storage. However, user initiated allocation can be done at any time. The user can also perform the same operations on labels which are intermixed with tables in storage.

There are three acquisition policies that can be followed.

1. Adding information about the contents of the data file to the index. This involves writing what we call a dictionary for the new file and supplying it to the system for the purpose of updating the index.
2. Writing the dictionary as in 1, but this time letting the system rewrite the tape into a form which can be read faster at the same time it updates the index.
3. Adding the file without a dictionary. This would require the user to write a dictionary for any variables used each time he came to the system (the individual user could save his own dictionary parts for future use). However, this would require another user of the same tape to duplicate efforts and most important would not allow for updating of the index which makes it difficult or impossible for future users to find the data. This type of action only makes sense in the case of a private file which is not to be added to the library and is to be read only once or twice.

The normal acquisitions procedure will include writing the dictionary and rewriting the file into a packed binary form. This can be done automatically by the system once the librarian has supplied the dictionary, and at very little added cost if it is done at the same time as the initial error checking. It is much faster to unpack a word into several parts than it is to read even a single additional word. The time saved by reading fewer words and not having to convert BCD to binary on each pass of the file will make up costs of the original rewriting and error checking in approximately three passes of a tape file.

Once a file has been added to the library, the user refers to the variables by name. He need never concern himself either with the location of the variable in the logical record or its packing. These are taken care of by the file dictionary, which is added to the system by the librarian in much the same way a card for a new book is added to a catalogue.

The cost of modifying the index structure is very low compared to the cost of modifying a data structure, and in fact the index is modified continually and automatically (heuristically) as determined from the present uses of the system.

The especial value of our computer system does not reside in its parts. We think that those we made work well, but they are not very novel. The system is important because it implements, not the application of the computer to the specifics of research problems, but the availability of the computer as a general instrument in a research laboratory. This difference is large and rests in the specifications of availability and instrumentality. Consider the automobile. It is immediately available for a wide variety of purposes, and one can be a good driver without understanding anything of its operating principles. The computer in a research laboratory should do as well, as easily, as often.