MIADS2

An Alphanumeric Map Information Assembly and Display System for a Large Computer

Elliot L. Amidon

U. S. FOREST SERVICE RESEARCH PAPER PSW- 38

Pacific Southwest
Forest and Range Experiment Station
Berkeley, California

1966

Forest Service - U. S. Department of Agriculture

Glossary

Block: a map or portion thereof; a unit for which tables will be produced; block contains one or more strips.

Card: an 80-column punch card.

Cell: the map area assigned a code (analogous to one dot in a dot grid); cell is a rectangle one-fifth inch (measured horizontally) by one-sixth inch, or one-thirtieth square inch.

Code: (1) integer system--unsigned, nonzero, two-digit integer (i.e., a number between 01 and 99 inclusive); (2) alphanumeric system--two characters.

Code System: a list of up to 98 code numbers (integer system) or 2,205 codes (alphanumeric system) and their definitions. Code 99, used for boundaries and other special purposes discussed elsewhere in this report, is ignored in all computations.

Line: a horizontal series of cells containing codes or blanks which are punched into one card.

MIADS: Map Information Assembly and Display System (integer).

MIADS2: Map Information Assembly and Display System (alphanumeric).

Source Map: a map, photo mosaic, or similar data source.

Strip: rows of lines made by listing ("printing") cards. Strip width is fixed at 7.2 inches (owing to use of 36 x 2 = 72 card columns). Strips are fastened together on their vertical edges to make an overlay.

Amidon, Elliot L.

1966. **MIADS2 ... an alphanumeric map information assembly and display system for a large computer.** Berkeley, Calif., Pacific SW. Forest & Range Exp. Sta. 12 pp., illus. (U.S. Forest Serv. Res. Paper PSW-38)

A major improvement and extension of the Map Information Assembly and Display System (MIADS) developed in 1964 is described. Basic principles remain unchanged, but the computer programs have been expanded and rewritten for a large computer, in Fortran IV and MAP languages. The code system is extended from 99 integers to about 2,200 alphanumeric 2-character codes. Hand-coding is reduced, and job preparation simplified. Total cost of using MIADS2 is about the same as that of using MIADS.

582:U681.3

Amidon, Elliot L.

1966. **MIADS2 ... an alphanumeric map information assembly and display system for a large computer.** Berkeley, Calif., Pacific SW. Forest & Range Exp. Sta. 12 pp., illus. (U.S. Forest Serv. Res. Paper PSW-38)

A major improvement and extension of the Map Information Assembly and Display System (MIADS) developed in 1964 is described. Basic principles remain unchanged, but the computer programs have been expanded and rewritten for a large computer, in Fortran IV and MAP languages. The code system is extended from 99 integers to about 2,200 alphanumeric 2-character codes. Hand-coding is reduced, and job preparation simplified. Total cost of using MIADS2 is about the same as that of using MIADS.

582:U681.3

CONTENTS

The Author

ELLIOT L. AMIDON is studying problems in production economics, with particular emphasis on multiple-use problems in forest lands. Native of Washington, D. C., he earned a bachelor's degree (1954) in forest management at Colorado Agricultural and Mechanical College. Awarded a Fulbright Grant in 1955, he spent a year at the University of Helsinki, where he studied forestry in Finland. He received a master's degree (1960) in agricultural economics from the University of California. Assigned to the Pacific Southwest Station's Berkeley staff since 1959, he has also served at the Lake States and Northeastern Forest Experiment Stations.

In 1964, the development of the Map Information Assembly and Display System (MIADS) was first announced (Amidon 1964). Since then, resource planners have found it useful in attacking a wide variety of land-oriented problems. But in the ensuing two years, it became apparent that what was needed was a more encompassing system--one that took advantage of the latest advancements in computer science. The result is a greatly expanded and improved system we call MIADS2. Basically unchanged is the original aim of a computer-oriented system--to record, update, assemble, and display map information quickly and efficiently. The principles and definitions, first published in 1964, still apply. The integer-computer programs used in MIADS, although still useful, are unsophisticated when compared to the programs used in MIADS2. Two major reasons underlie the need for an improved system.

First, MIADS was restricted to a 2-integer code system. Some tasks appeared too large at first glance, but could be reduced if very careful planning preceded hand coding. For many problems a 99-code system was simply too small. There was not much "slack" for adding codes, particularly if the need for them arose half-way through a massive coding job. The conversion to the character system in MIADS2 makes about 2,200 codes available. This not only permits formidable problems to be tackled, but perhaps as important, adds mnemonic value to the code system. For example, whereas "grass" was "03" before, it became "GR." Actually, not all of the alphanumeric codes are equally useful; equal signs (====) may be fine for roads, but what good is -) or /)?

The other major reason is more subtle but nearly as important. The programs were designed to run on either a small decimal computer or a large binary machine. The end result reflected restrictions which were primarily imposed by the smaller processor.

A host of lesser reasons motivated us to extend the original system. One of them is always present --technological change in both computers and their languages. Much of the data preparation work performed by tabulating equipment could be shifted to the computer. Similarly, some of the user's job setup time could be saved by using control cards with English words. The cards could be pre-punched and their functions easily remembered. Experience indicated that some options should be dropped-particularly cumulative tables --while others--like tape handling capability-- should be added. Finally, we realized that hand coding, the most costly part of any job, could be reduced by letting the computer fill in "patches" of repetitive codes.

This paper describes MIADS2, compares differences between it and MIADS, and explains its new features. Table 1 summarizes the differences between the two systems.

Characteristics of the System

Codes

According to combinatorial theory, the maximum size of a code system is reached by raising the total number of characters to a power. The power, or exponent, is simply the number of characters per code. For example, if only three characters were available--A, B, and C-- a two-character code system would contain three squared, or nine, unique codes--AA, AB, ... , etc. For the same character set, a three-character code would contain more information-three cubed, or 27. items. We decided that a two-character code would be enough for attacking most problems and allow, given a common print spacing, a nearly-square grid cell.

The integer system used in MIADS consisted of

Table 1.--*Comparison of features of MIADS and MIADS2*

Item	MIADS	MIADS2
Theoretical code system size	100	2304
Processor	Small or large, e.g. IBM 1620.7090,7094	Large (32K, e.g. IBM 7040, 7090.7094
Language	Fortran II or IV	Fortran IV and MAP
Speed (large processor)		
1. Mapping program (thousand map cards)	3/4 min.	3/4 min.
2. Combinations program (thousand card pairs)	1-2 min.	1.5-5 min.
Input media	Cards	Cards, magnetic tape
Tab room equipment	Several machines	Keypunch and verifier
Code systems combined	2-4	2
Map card data preparation	Hand coding (all cells)	Hand coding, computer fills cells
Data editing	Reproducer (e.g. IBM 514)	Computer
Types of tables	13	9
Control card		
1. Parameters	Numeric	Alphanumeric
2. Sequence	Fixed	Variable
Number of blocks	Limited	Unlimited

ten characters-- 0 to 9. It allowed a maximum of 100 codes in the system. Actually, 98 were available for manipulation because zero-zero (00) was equated to blank (nothing) and 99 was reserved for a boundary code. The alphanumeric system used in MIADS2 consists of 48 characters available in the printer set, or a theoretical maximum of 2,304 codes. But for a variety of reasons (discussed in the operating manual) about 100 codes were excluded from use on map data cards. Some printers have 64 characters (allowing 4,096 codes), and the current programs will accept up to 64, with minor modifications.

The same combinatorial principles apply to combinations from two or more code systems. Assume that mapped information is coded into two 98-integer code systems. The combinations program has to search for 1 to 98 pairs of two-character codes out of the 98-squared, or 9,604 possibilities. With the alphanumeric system, the number of possible combinations rises to about 5 million.

Computer Programs

MIADS2, like MIADS, contains two mainline programs: (a) the mapping program (MAP2), and (b) the combinations program (COMBN2). Both contain Fortran IV and MAP subprograms developed and tested on the DCS monitor system at the University of California Computer Center, Berkeley, California.[1] With version 13 of Fortran IV and a 32K IBM 7094 computer, the mapping and combinations programs will reside in memory with. respectively, 652 and 9,000 words left over. This particular storage allocation should not be considered as "best." It is merely the end result of a long series of compromises among speed, storage, and program options. The programs are segmented into many smaller routines or "building blocks" on the assumption that users may have different criteria and will want to modify their programs accordingly.

Data Preparation

The basic data collection procedure has not changed. MIADS (integer) map data input can also be processed by MIADS2. The user sets up an implicit coordinate system by putting strips of grid paper over a map and hand coding the desired information into cells. There is, however, one difference--he need not put a code in every cell. Often there will occur a "patch" of several grid cells, all of which are to be assigned the same code. Previously the coder or the keypunch operator wrote the code in every cell. The computer will now do this copy work if the coder outlines the border of the "patch" with the code (fig. 1)--he need only outline the left-most edge. Verification of hand coding work is easier if the entire border is filled. But this time-saving advantage has its

[1] The MAP subprograms were provided by Robert M. Russell, programer at the Pacific Southwest Forest and Range Experiment Station. Berkeley. Calif.

```
A
       0606                          0606                     1

     06    06                      06060606                   2
   06      06                    0606060606                   3
   06      06                    0606060606                   4
     060606                        060606                     5

 OUTLINED PATCH                HAND OR KEYPUNCH FILLING
```

```
B
  --INPUT--                              --OUTPUT--            --LINE NO.--


       0606ABABABAB99                 0606ABABABAB99                1
     06      06AB      AB99         06060606ABABABAB99              2
   06        06          AB99     060606060606060606AB99            3
   06        06AB        AB99     0606060606ABABABABAB99            4
     060606AB            AB99       060606ABABABABABAB99            5
           AB        AB99                 ABABABABAB99              6
             ABABAB                         ABABABABABABABABAB      7

  OUTLINED PATCH                          COMPUTER FILLING

NOTE ERRORS IN LINES 3 AND 7 DUE TO BOTH A MISSING LEFT-HAND 'AB' AND
     A TERMINATING BOUNDARY CODE '99'.
```

Figure 1. -- *The job of hand-coding -- translating map characteristics into numbers -- is considerably reduced under MIADS2.* A, *cells were hand-filled under MIADS;* B, *the computer does the filling in under MIADS2, and only the left-most edge of each "patch" must be hand-filled.*

price since an omitted code can cause errors (fig. 1). The coder may fill a patch by hand since excess codes do not affect the filling process. The reduction in hand-coding cost attributable to the filling option is directly related to the size of the "patches." This option also makes very large patches feasible, changing the capability of the system to extract and manipulate map data.

Options and Output

Options

A major structural change for both the mapping and combinations program is the procedure for setting up a job. Previously all map records were counted before processing and control cards were inserted before blocks, in a precise fashion. The current control procedure allows control cards to be placed with the blocks of map records to which they apply.

A deck setup for a mapping program run can be illustrated only in a general way (fig. 2). Except for segmenting a deck into jobs and blocks, no regular pattern is needed. Control cards are inserted as needed. Obtaining useful results for even a single block of map records may require as few as two or a maximum of 461 control cards. Consequently, by using English or pidgin English words on control cards, deck setup is easier. Information on control card suboptions and other

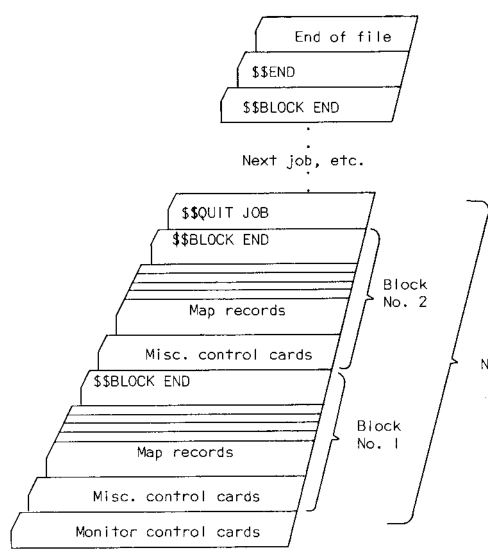

Figure 2--*MAP2--general deck setup for a run, showing block and job segmentation.*

details related to deck setup will be found in the operating manual. The deck setup for MIADS2 control card operators is as follows:

MIADS2 Control Cards (apply to both programs):

1. $$HEADER
2. $$TITLE
3. $$QUIT JOB
4. $$END
5. $$TAPE
6. $$LINES

Additional Control Cards for MAP2:

1. $$BLOCK END
2. $$SCALE
3. $$CODES REMOVED
4. $$RATES
5. $$FILL
6. $$TABLES
7. $$OVERLAY

Additional Control Cards for COMBN2:

1. $$CHANGE
2. $$EXCLUDED
3. $$COMBINE
4. $$CARDS
5. $$PRINT
6. $$PUNCH

Output

The types of tables printed out by MAP2 have been reduced from 13 to 9. Previously, acreages, products, and their corresponding proportions could be cumulated over codes within a block as well as accumulated over blocks. It was necessary either to anticipate cumulation when designing the code system or to change the codes around into the desired ascending order later. Such forethought was practical with the small integer code system, but not with the large one. Therefore, the cumulative option was dropped. Tables and other items that can be produced by MAP2 under MIADS2 are as follows:

Block Tables

Block Code:

1. Frequencies
2. Proportions
3. Acreages
4. Products
5. Product proportions

Block Items

1. Block frequency total
2. Block acreage total
3. Acreage represented by one grid cell
4. Scale in inches per mile
5. Block product total

Accumulated Block Tables

Accumulated Block Code:

1. Proportions
2. Acreages
3. Products
4. Product proportions

Accumulated Block Items

1. Total block acreage
2. Total block product.

Block acreage could be computed in four ways in MIADS. Because map scale--or more precisely, representative fraction--is always sufficient to calculate block area, the other three ways were dropped.

In MIADS, combinations could be found from two to four code systems simultaneously. The combinations program (COMBN2) still changes codes within one system, but will combine only two code systems. The explanation for this is the same as that given for reducing table output from MAP2. Combining several large systems at once requires too much advance planning. Besides, given a capability of combining two systems, any additional number may be combined by successive runs.

Various mixtures of printing and punching, or both, are permitted with MIADS2. Tape input/output is comparatively restricted in that only one tape job is permitted per run--one file in length; that is, tape writing or reading always starts at the beginning of a tape. This limitation implies that each large job (e.g., a coded National Forest map) will be assigned to one tape. If this procedure is inconvenient, simple programs can be written to store and manipulate the map tape records.

Time Estimates

Time to load the object deck is about a half-minute for either computer program.

Processing time for the mapping program is independent of the size of the code system. But it does vary with the number of map records and table output. Running time is essentially the same as previously stated for the integer code system on an IBM 7090 computer--$^{3}/_{4}$ minute per 1,000 map records. This same estimate applies to code changing with the combinations program.

Combining codes is somewhat more expensive than map program processing. A fixed amount of time is required to sort the combinations sought before processing the pairs of map records. Most jobs will involve less than a thousand combinations and a rule-of-thumb of $1^1/_2$ minutes per thousand pairs of records is accurate enough. For larger problems, a time estimation procedure is provided in the operating manual.

Total cost, the sum of all hand and computer work, appears to be unchanged from the original system. For large jobs, that is, 3,000 map records or more, the cost per acre should still be about 15/100 of a cent, based on map scale of 2 inches per mile.

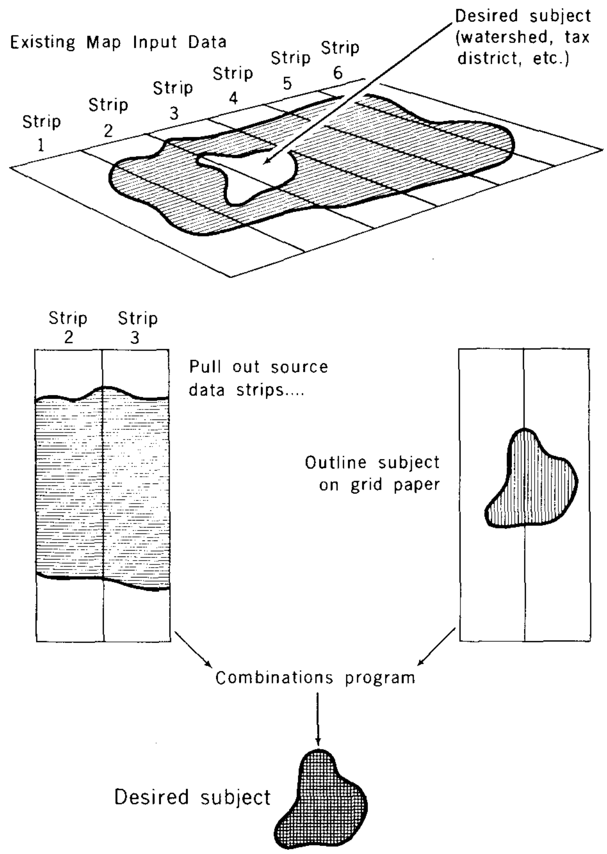

Figure 3.--*Filling and combining to "pull out" a portion of map data.*

Applications of the System

The major use of the system will continue to be for inventory type work. Of the less obvious uses, perhaps the most interesting is the ability to assemble, display, and update input data for analytical models. For example, a prototype model which integrates MIADS and linear programing into a larger computer-oriented system has been described recently (McConnen, Navon, and Amidon 1966;[2]).

The large code system now available will allow much larger and more complex applications than ever before. The addition of the filling option also adds a capability that, while possible before, would have required too much hand coding. Very large "patches" can be filled as readily as small ones. Consequently filling, followed by application of the combinations program, can be used to extract irregular blocks of map data cards from an existing code set (fig. 3). Low-cost filling also enables the user to extract alternative configurations of mapped data. He can "cut and try" alternative boundaries for compartments within a forest, or determine the effect of alternative rules on, say, recreation area boundaries (fig. 4) .[3]

Finally, a problem whose scope grows beyond

[2] Navon, Daniel I. *Computer-oriented systems for wildland management*. (MS. accepted by J. Forestry.)

[3] The fill option as a tool for making "policy maps" was first recognized by Dr. R. J. McCormen, formerly project leader of production economics, Pacific Southwest Forest and Range Experiment Station, Berkeley, Calif.

the capability of an integer code system, or becomes too costly on a small computer, can be transferred to MIADS2 without any difficulty.

The examples in the appendix, along with the imagination of the user, will suggest potential applications of MIADS2.

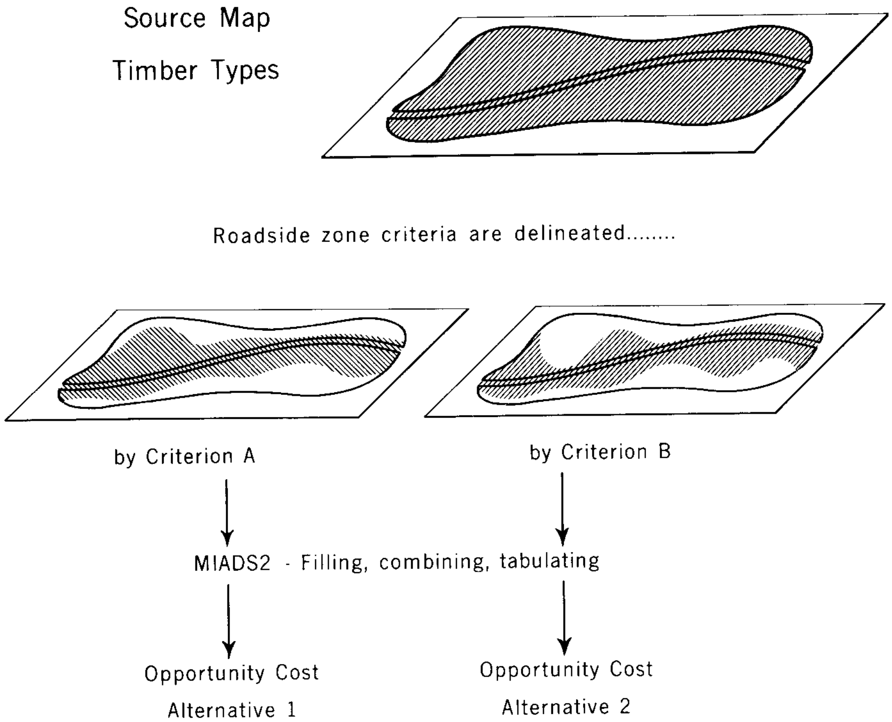

Figure 4.--*The fill option helps produce "policy maps." In this example two compartment boundaries are defined.*

Future Development

The point at which to end development work is seldom clearcut. The MIADS system is no exception. At this point, we wish to describe some revisions and extensions of MIADS2 that show promise. Generally, the proposed modifications were not judged worth incorporating into the system because they lack general application. For some users, however, their development may be quite worthwhile under their particular circumstances. The suggested changes fall into temporal order--improving the old integer system, modifying the present alphanumeric programs, and pointing out some promising extensions.

Integer System

Mapping Program

No particular problem has been encountered to date with the mapping program in MIADS, but the IBM 1620 computer programs do require a 60K memory. And since 40K is much more common, the program should have been segmented so it could be run in two passes. In MIADS one minor change has been made, not in the program, but to the reproducer wiring diagram. Listing will suppress all zeros, as before, except the nine integers with trailing zeros, i.e., codes 10, 20, … , 90.

Combinations Program

The IBM 1620 computer program fits 40K but is so slow as to be impractical except for small numbers of combinations or a couple hundred map records. Processing is faster if the most frequently occurring combinations are at the beginning of the list. The absolute expected frequency will seldom be known, but even a relative ranking will help.

Two alternative algorithms were developed. The first, for a 60K memory, was about 10 times faster than the published program. The second method, which would fit a 40K memory, was $4^1/_2$ times faster than the published routine. This gain, although substantial, was not enough to justify program revision. For large jobs the user should switch to the large computer version.

The control procedure specified prevented more than one input combination from having the same output code. This restriction was unnecessary from the standpoint of the combining algorithm .[4] Its effect is not obvious and is best illustrated by an example.

Suppose several forest land classes are uniquely coded for multiplication by such rates as growth per acre. Later, for some other alternative use, such as recreation or grazing, these categories can be lumped. They cannot be all assigned the same output code by this combination program. A clumsy way around this problem would be to change codes in each system before combining. This change, however, should not be necessary, and is not-with MIADS2.

Alphanumeric System

In MIADS2, both MAP2 and COMBN2 are basically character-handling routines. Generally speaking, Fortran is more suitable for computational work than character manipulation. For as Golden (1965, p. 153) has reported: "It can be claimed with some justification that Fortran is far from ideal as a character manipulation language." Assuming a trend toward languages, as well as hardware, of a more general nature, then character-handling in the future should become easier. This trend would reduce the effort, say, of writing MIADS-type subroutines, allowing code system size to be specified for the particular problem at hand.

The computer time required by MIADS2 appears to be reasonably low. However, it can be reduced further, perhaps a fourth or more, by modifying the input-output subprograms. The MAP2 and COMBN2 programs process data independently, in the sense that they have their own data conversion routines. Because the output of one program is often input for another, time is spent converting external codes to internal ones used by the processor, and back. A programer can reduce conversion time, for example, by storing the internal codes on tape, ready for input to the next run. He then must keep close track of the data flow, but it might be worthwhile for large jobs.

The combinations program is based on the binary search method. It requires that the combinations sought be sorted. The set-up time for sorting is only three seconds for 500 combinations, but rises to about 211 seconds for 2,299 combinations. Sorting is done by a modified version of the exchange method. Programers generally agree that no one method of sorting is always best. If very large numbers of combinations are to be sorted often, however, another method should be considered. There are many alternative sorting algorithms (Gotlieb 1963).

Adaptations and Extensions

One of the advantages of the original integer system was its adaptability. A given set of data could be processed on a small, variable-word-length computer (IBM 1620) or a large, fixed-word-length computer (e.g., IBM 7090, 7094). MIADS recently has been adapted to both an IBM 7040 and an IBM 360 computer, model 30.[5] Although computer time cost was always less on the larger machine, total cost of a job might not be, because of accessibility or other circumstances. Ideally, the same alternatives should still be available. They are not, however, largely because of reasons mentioned earlier. These reasons may be grossly summarized by stating that the smaller processor places too many restrictions on the flexibility of the larger system; in addition, the disparity in processing cost becomes even larger. Therefore, we put almost all our resources into a large processor system.

MIADS2, however, can be extended to processors of various sizes and capabilities because of its structure. Each program is a collection of many smaller subprograms. A programer can modify, add, or delete them to suit his circumstances. For example, by using the same techniques incorporated in MIADS2, we were able to write an alphanumeric mapping program in Fortran II for an IBM 1620 computer. It was deliberately restricted to 40,000 locations, although the computer available had 60K. The program had few options, but the control and map data cards would be accepted without change by MIADS2. The routine will fill and process one block of map cards, producing a single overlay and acreage table. The maximum number of codes permitted in the system is 729. After trying a number of input-output configurations, throughput was raised to eight full

[4] This limitation was uncovered by R. E. England in the course of applying MIADS as part of his thesis work under the direction of Professor Louis Hamill, Department of Geography. University of Alberta, Calgary, Alberta, Canada.

[5] Personal correspondence with C. L. Kirby, Forestry Branch, Department of Forestry. Calgary, Alberta, Can-ada. Sept. 9, 1966.

data cards per minute. This speed is only half that of the original integer program, and it had many more options. Since this was an exploratory effort, machine language coding was not justified. Installations with disks and the capability of mixing Fortran and machine language subroutines may be able to triple processing speed. Although the program is relatively crude, it does demonstrate that the MIADS2 techniques can be adapted and extended to smaller processors.

Appendix

A. Examples of Computer Input/Output

1. A comparison of MIADS and MIADS2 mapping program input based on a simplified example.

The arithmetic behind the mapping program tables is described in U.S. Forest Service Research Paper PSW-17 (Amidon 1964, pp. 24-28). The MIADS integer input for the simplified example on page 27 of that publication is repeated below. Following it is the corresponding MAP2 input.

MIADS DECK SETUP

```
    1
 DERIVATION OF MAPPING PROGRAM TABLES. NUMERICAL EXAMPLES.  11/22/63
    1     2     3     1     2     3 2
 3
 1        100
 1       1000
      4E-1       5E-1       6E-1
010101020202020203O3                                                    2 1     1
010101010202020303O3                                                    1 2     2
```

MIADS2 DECK SETUP

```
$$HEADER              1
$$TITLE DERIVATION OF MAPPING PROGRAM TABLES. NUMERICAL EXAMPLES. 11/22/63
$$SCALE           43380
$$CODES REMOVED
03
$$TABLES              2
$$OVERLAY             1
$$RATES
01     4E-102       5E-103       6E-1
010101020202020203O3                                                          1
$$BLOCK END
$$SCALE          137178
$$TABLES              1
010101010202020303O3                                                          2
$$BLOCK END
$$END
```

2. A MIADS2 job consisting entirely of control cards.

By presenting an extreme example, the variation possible with a MAP2 job setup is shown. The input listed below consists entirely of control cards. The program output is also shown.

```
$$TITLE    NOTE- THIS JOB CONSISTS ENTIRELY OF CONTROL CARDS.
$$TITLE GRID CELL AREA IN ACRES AND SCALE IN INCHES PER MILE FOR VARIOUS
$$TITLE     MAP REPRESENTATIVE FRACTIONS
$$SCALE             5280  OR 12 INCHES PER MILE
$$SCALE            12000
$$SCALE            13718
$$SCALE            15840  OR  4 INCHES PER MILE
$$SCALE            23760
$$SCALE            31680  OR  2 INCHES PER MILE
$$SCALE            43380
$$SCALE            63360  OR  1 INCH   PER MILE
$$SCALE           137178
$$SCALE           999999,THE LIMIT FOR THIS PROGRAM--NEARLY 1 MILLION.
$$QUIT JOB
```

```
                    FORTRAN 4 MAPPING PROGRAM, MIADS2 SYSTEM, 1966

          QUESTIONS CONCERNING PROGRAM MODIFICATION SHOULD BE DIRECTED TO

               ELLIOT L. AMIDON
               P.S.W. FOREST AND RANGE EXPT. STA.
               P.O. BOX 245
               BERKELEY, CALIF

    NOTE- THIS JOB CONSISTS ENTIRELY OF CONTROL CARDS.

  GRID CELL AREA IN ACRES AND SCALE IN INCHES PER MILE FOR VARIOUS

     MAP REPRESENTATIVE FRACTIONS

$$SCALE             5280  OR 12 INCHES PER MILE

                          INCHES PER MILE =  0.12000000E 02

                          ONE CELL(1/30TH SQ.IN.) =  0.14814815E 00 ACRES

$$SCALE            12000

                          INCHES PER MILE =  0.52800000E 01

                          ONE CELL(1/30TH SQ.IN.) =  0.76522803E 00 ACRES

$$SCALE            13718

                          INCHES PER MILE =  0.46187491E 01

                          ONE CELL(1/30TH SQ.IN.) =  0.10000230E 01 ACRES

$$SCALE            15840  OR  4 INCHES PER MILE

                          INCHES PER MILE =  0.40000000E 01

                          ONE CELL(1/30TH SQ.IN.) =  0.13333333E 01 ACRES
```

```
$$SCALE          23760
                          INCHES PER MILE =  0.26666667E 01
                          ONE CELL(1/30TH SQ.IN.) =  0.30000000E 01 ACRES
$$SCALE          31680  OR  2 INCHES PER MILE
                          INCHES PER MILE =  0.20000000E 01
                          ONE CELL(1/30TH SQ.IN.) =  0.53333333E 01 ACRES
$$SCALE          43380
                          INCHES PER MILE =  0.14605809E 01
                          ONE CELL(1/30TH SQ.IN.) =  0.10000172E 02 ACRES
$$SCALE          63360  OR  1 INCH   PER MILE
                          INCHES PER MILE =  0.10000000E 01
                          ONE CELL(1/30TH SQ.IN.) =  0.21333333E 02 ACRES
$$SCALE         137178
                          INCHES PER MILE =  0.46188164E 00
                          ONE CELL(1/30TH SQ.IN.) =  0.99999380E 02 ACRES
$$SCALE         999999,THE LIMIT FOR THIS PROGRAM--NEARLY 1 MILLION.
                          INCHES PER MILE =  0.63360063E-01
                          ONE CELL(1/30TH SQ.IN.) =  0.53140729E 04 ACRES
$$QUIT JOB
```

3. A demonstration of code filling and removal.

The input listed below can be processed by either MAP2 or the exploratory IBM 1620 program. The 1620 output also is shown.

```
$$TITLE  1620 TEST DATA
$$TITLE            A DEMONSTRATION OF CODE FILLING AND REMOVAL
$$TITLE EXAMPLE IS A 'CHECKERBOARD' OF PUBLIC AND PRIVATE LAND, WITH
$$TITLE   OWNERSHIP ALTERNATING IN EACH SQUARE MILE(640 ACRES).
$$SCALE            63360 NOTE- ONE SQUARE INCH EQUALS 640 ACRES
$$TITLE THE FIRST STEP IS TO FILL UP THE EMPTY CELLS WITH CODES
$$TITLE THE SECOND STEP IS TO MAKE FEDERAL LAND OVERLAY BY REMOVING P'S
$$FILL
$$CODES REMOVED         READING STOPS WHEN A PAIR OF BLANKS IS ENCOUNTERED.
P2P3P4P5P6P7P8
99999999999999      99999999999999999999999999999999999999999999           1
99F1        99      99F2        P2        F3        P3        99           2
99F1        99      99F2        P2        F3        P3        99           3
99F1        99      99F2        P2        F3        P3        99           4
99F1        99      99F2        P2        F3        P3        99           5
99F1        99      99F2        P2        F3        P3        99           6
99F1        9999999999F2        P2        F3        P3        99           7
999999999999F4      P4          F5        P5        F6        99           8
            99F4    P4          F5        P5        F6        99           9
            99F4    P4          F5        P5        F6        99          10
            99F4    P4          F5        P5        F6        99          11
            99F4    P4          F5        P5        F6        99          12
999999999999F4      P4          F5        P5        F6        99          13
99F7        P6      F8          P7        F9        P8        99          14
99F7        P6      F8          P7        F9        P8        99          15
99F7        P6      F8          P7        F9        P8        99          16
99F7        P6      F8          P7        F9        P8        99          17
99F7        P6      F8          P7        F9        P8        99          18
99F7        P6      F8          P7        F9        P8        99          19
99999999999999999999999999999999999999999999999999999999999999          20
$$END
```

```
    1620 TEST DATA
            A DEMONSTRATION OF CODE FILLING AND REMOVAL
   EXAMPLE IS A 'CHECKERBOARD' OF PUBLIC AND PRIVATE LAND, WITH
     OWNERSHIP ALTERNATING IN EACH SQUARE MILE(640 ACRES).
  $$SCALE            63360 NOTE- ONE SQUARE INCH EQUALS 640 ACRES
  INCHES PER MILE=  .10000000E+01 ONE CELL=  .21333333E+02 ACRES
   THE FIRST STEP IS TO FILL UP THE EMPTY CELLS WITH CODES
   THE SECOND STEP IS TO MAKE FEDERAL LAND OVERLAY BY REMOVING P'S
  $$FILL
FILL INPUT CARDS
  $$CODES REMOVED         READING STOPS WHEN A PAIR OF BLANKS IS ENCOUNTERED.
REMOVE   7 CODES
99999999999999      99999999999999999999999999999999999999999999           1
99F1F1F1F1F199      99F2F2F2F2F2          F3F3F3F3F3          99           2
99F1F1F1F1F199      99F2F2F2F2F2          F3F3F3F3F3          99           3
99F1F1F1F1F199      99F2F2F2F2F2          F3F3F3F3F3          99           4
99F1F1F1F1F199      99F2F2F2F2F2          F3F3F3F3F3          99           5
99F1F1F1F1F199      99F2F2F2F2F2          F3F3F3F3F3          99           6
99F1F1F1F1F19999999999F2F2F2F2F2          F3F3F3F3F3          99           7
999999999999F4F4F4F4F4          F5F5F5F5F5          F6F6F6F6F699           8
            99F4F4F4F4F4        F5F5F5F5F5          F6F6F6F6F699           9
            99F4F4F4F4F4        F5F5F5F5F5          F6F6F6F6F699          10
            99F4F4F4F4F4        F5F5F5F5F5          F6F6F6F6F699          11
            99F4F4F4F4F4        F5F5F5F5F5          F6F6F6F6F699          12
999999999999F4F4F4F4F4          F5F5F5F5F5          F6F6F6F6F699          13
99F7F7F7F7F7        F8F8F8F8F8            F9F9F9F9F9          99          14
99F7F7F7F7F7        F8F8F8F8F8            F9F9F9F9F9          99          15
99F7F7F7F7F7        F8F8F8F8F8            F9F9F9F9F9          99          16
99F7F7F7F7F7        F8F8F8F8F8            F9F9F9F9F9          99          17
99F7F7F7F7F7        F8F8F8F8F8            F9F9F9F9F9          99          18
99F7F7F7F7F7        F8F8F8F8F8            F9F9F9F9F9          99          19
99999999999999999999999999999999999999999999999999999999999999          20
  $$END
                    BLOCK CODE ACREAGES
  ROW    1 CODES                F1              F2              F3              F4
  CUM.COUNT    4  .63999999E+03  .63999999E+03  .63999999E+03  .63999999E+03
  ROW    2 CODES                F5              F6              F7              F8
  CUM.COUNT    8  .63999999E+03  .63999999E+03  .63999999E+03  .63999999E+03
  ROW    3 CODES                F9              P2              P3              P4
  CUM.COUNT   12  .63999999E+03  .63999999E+03  .63999999E+03  .63999999E+03
  ROW    4 CODES                P5              P6              P7              P8
  CUM.COUNT   16  .63999999E+03  .63999999E+03  .63999999E+03  .63999999E+03
BLOCK ACREAGE=   .10239998E+05
  INPUT-ALL RECORDS  31 ALL MAP RECORDS  20 ALL CONTROL CARDS  11
NO.OF BAD MAP RECORDS=   0 **END OF RUN**
```

B. Availability of Operating Manual

Detailed instructions on how to use MIADS2 and for preparing and processing data by the computer programs are included in an operating manual titled "MIADS2." This manual may be obtained as an interlibrary loan by writing to-

Computer Services Librarian
Pacific Southwest Forest and Range
Experiment Station
P. O. Box 245
Berkeley, California 94701.

Selected References

Amidon, Elliot L.
1964. **A computer-oriented system for assembling and displaying land management information.** U.S. Forest Serv. Res. Paper PSW-17. Pacific SW. Forest & Range Exp. Sta., Berkeley, Calif. 34 pp., illus.

Bengtsson, Bengt-Erik, and Nordbeck, Stig.
1964. **Construction of isarithms and isarithmic maps by computers.** BIT (Nordisk Tidskrift for Informationsbehandling) 4(1964): 87-105.

Brown, C. E.
1964. **A machine method for mapping insect survey records.** The Forest. Chron. 4(40): 445-449.

Clawson, Marion.
1966. **Recent efforts to improve land use information.** J. Amer. Stat. Ass. 3 15(61): 647-657.

Department of Forestry.
1965. **The Canada land inventory.** Dep. Forestry Publ. 1088, Rep. 1. Ottawa, Canada. 12 pp.

Golden, James T.
1965. **Fortran IV programming and computing.** 270 pp., illus. Englewood Cliffs, New Jersey: Prentice-Hall.

Gotlieb, C. C.
1963. **Applications of digital computers.** pp. 68-84. Walter F. Freiberger and William Prager, eds. Waltham, Mass.: Blaisdell.

McConnen, R. J., Navon, D. I., and Amidon, E. L.
1966. **Efficient development and use of forest lands--an outline of a prototype computer-oriented system for operational planning.** Forest. Comm. Forest Rec. 59, pp. 18-32. London: Her Majesty's Stationery Office.

McCormack, R. J.
1966. **The Canada land inventory of ARDA.** The Forest. Chron. 1(42): 45-50.

Monmonier, Mark S.
1965. **The production of shaded maps on the digital computer.** The Prof. Geogr. 5(17): 13-14.

Perry, Benson, and Mendelsohn, Mortimer L.
1964. **Picture generation with a standard line printer.** Commun. of the ACM 5(7): 311-313.

GPO 973-775