UNIT 35 - RASTER STORAGE

UNIT 35 - RASTER STORAGE

Compiled with assistance from Donna Peuquet, Pennsylvania State University

The latter portion of this unit and the following two require familiarity with numbering systems in base 2 and 4 as well as techniques for conversion between these and decimal. You may wish to provide your students with some background material on these topics before tackling these units.

UNIT 35 - RASTER STORAGE

Compiled with assistance from Donna Peuquet, Pennsylvania State University

A. INTRODUCTION

Why use raster?

- data are acquired in that form from remote sensing, photogrammetry or scanning
- is a common way of structuring digital elevation data
- raster assumes no prior knowledge of the phenomenon, sampling is done uniformly
    - knowledge of variability would allow us to sample more heavily in areas of high variability (rugged terrain) and less heavily in smooth terrain
- data are often converted to raster as a common format for data interchange
- for merging with remote sensing images or DEMs
- raster algorithms are often simpler and faster
    - e.g. buffer zone generation is simpler in raster
- raster may be appropriate if the solution requires uniform resolution, e.g. in finding optimum routes for linear features such as power lines, or in inferring the locations of stream networks from DEMs

Objectives

- there are many options for storing raster data (many data structures)
- some are more economical than others in use of storage
- some are more efficient in access and processing speed
- this unit looks at some of the options and issues involved
- many of these issues were introduced in Unit 4, they are expanded upon here

B. STORAGE OPTIONS FOR RASTER DATA

- by convention, raster data is normally stored row by row from the top left
    - this is the European/North American reading order
    - is also the order of scan of a TV image
- example
- the image A A A A A B B B A A B B A A A B
    - would be stored in 16 memory positions, one for each pixel, in the sequence:

      A A A A A B B B A A B B A A A B

What if there is more than one layer?

- two options:

  1. store the layers separately

     - this is the normal practice

  2. store all information for each pixel together
     - this requires extra space to be allocated initially within each pixel's storage location for layers which might be created later during analysis
     - this is usually difficult to anticipate

- (note in remote sensing, these concepts are called "band sequential" and "band interleaved by pixel" respectively)

What do raster systems store in each pixel?

- some allow only an integer, in a fixed range, e.g. -127 to +127 (1 byte per pixel) or -32767 to +32767 (2 bytes per pixel)

- some allow integers, real (decimal) numbers and mixed alphabetic letters and numbers in each pixel
  - in this case it helps if the system keeps track of what type of data is stored in each layer and stops the user doing wrong types of analysis on the data
  - example:
    - vegetation data is recorded as a class (A thru G) in each pixel
    - elevation data is recorded as a decimal number (e.g. 100.3 m)
    - the system should not allow the user to add the pixel values from the two layers (A + 100.3) or perform any other kind of arithmetic operation on the vegetation data

Raster/Vector combinations

- many raster-based systems allow vector input Example:
  - a polygon, defined by its vertices, is input
  - convert this to a raster
    - e.g. assign 1 to all pixels inside the polygon, 0 to all outside

- some forms of data are really hybrids of raster and vector:
  - Freeman chain code has finite resolution based on pixels (raster-like) but defines lines and the boundaries of objects (vector-like)
  - a raster can be used to define objects at fixed resolution if every pixel is given an object number instead of a value
    - the object numbers are pointers to an attribute table:

      Raster ObjectAttributes

      23 23 23 24 23 A 100.0 23 23 24 24 24 B 101.1 23 23 24 24 23 23 23 24

  - this gives us an object with its attributes, plus a list of pixels associated with the object instead of the object's coordinates

- in this sense, a raster is a finite resolution geometry rather than an alternative way of structuring spatial data

C. RUN ENCODING

- geographical data tends to be "spatially autocorrelated", meaning that objects which are close to each other tend to have similar attributes
  - Tobler expressed it this way: "All things are related, but nearby things are more related than distant things"

- because of this principle, we expect neighboring pixels to have similar values
  - so instead of repeating pixel values, we can code the raster as pairs of numbers - (run length, value)
    - e.g. instead of 16 pixel values in original raster matrix, we have:

      4A 1A 3B 2A 2B 3A 1B

    - produces 7 integer/value pairs to be stored

- if a run is not required to break at the end of each line we can compress this further:

  5A 3B 2A 2B 3A 1B = 6 pairs

- however, it helps to limit the possible size of the run so that we can use less space to store the run length, as the amount of space allocated must be sufficient for the maximum run length

Problems

- layers now have different lengths depending on the amount of compression (lengths of runs)

- storing all layers together for each pixel now makes no sense

- run encoding would be little use for DEM data or any other type of data where neighboring pixels almost always have different values

D. SCAN ORDER

1. Row order

- described already

- are there better ways of ordering the raster than row by row from the top left?
  - other orders may produce greater compression

overhead/handout - Standard scan orders

2. Row prime order (Boustrophedon)

- suppose we reverse every other row:

  diagram

- this has the charming name boustrophedon from the Greek for "how an oxen plows a field"

- avoids a long jump at the end of each row, so perhaps the raster would produce fewer runs and thus greater compression

- this order is used in the Public Land Survey System: the sections in each township are numbered in this way

- one the original raster (page 35-3) it results in:

  4A 3B 3A 3B 3A = 5 runs

3. Morton order

overhead/handout - (cont) Standard scan orders

- Morton order is the basis of many efforts to reduce database volume
  - named for Guy Morton who devised it as a way of ordering data in the Canada Geographic Information System
  - however, this way of ordering or scanning a raster was well known long before Morton
    - it is associated with the names of several mathematicians and geometers: Hilbert, Peano, and Koch
  - coincidentally, Morton is the name of the lower left corner county in Kansas

- the strategy is to exhaust each area of the map in sequence, whereas row by row order scans from one side to the other
  - this minimizes the number of large jumps

  diagram

- this is one of several hierarchical ordering systems
  - it is built up level by level, repeating the same pattern at each level, as follows

    2 3 10 11 14 15 42 43 46 47 58 59 62 63 0 1 8 9 12 13 40 41 44 45 56 57 60 61 2 3 6 7 34 35 38 39 50 51 54 55 0 1 4 5 32 33 36 37 48 49 52 53 10 11 14 15 26 27 30 31 8 9 12 13 24 25 28 29

    2 3 6 7 18 19 22 23 0 1 4 5 16 17 20 21

it is only valid for square arrays where the numbers of rows and columns are powers of 2

  - e.g. 2x2, 4x4, 8x8, 16x16, 32x32, 64x64, etc.

- how does it do on our 4x4 array?

  5A 3B 1A 1B 2A 2B 2A = 7 runs

  - which is as long as row by row compression

4. Peano scan (also Pi-Order or Hilbert)

- the Peano scan or Pi-order is like boustrophedon in always moving to a neighboring pixel

  diagram

  - the name Peano is associated with both this and Morton orders, though more often with this

- it is also hierarchical, but the pattern appears in different orientations at different levels

Comparing scan orders

- it is useful to look at a comparison of the compression rates obtained by the different orders (see Goodchild and Grandfield, 1983)

  overhead/handout - Scan order comparison

- the comparison shown used a number of 64x64 pixel images:
  - all pixels are either B or W
  - vary from:
    - images with large patches of black or white to
    - very chaotic images in which each pixel is independently black or white
  - in the table, H indicates the amount of chaos in the image:
    - the higher H, the larger the patches
    - low H corresponds to chaotic images
  - each line in the table gives the numbers of runs required to code the same black and white image
  - the values in the last line were calculated theoretically

- this table shows that scan order makes little difference to data compression
  - the number of runs is not greatly affected by scan order for a given image

  - however, the orders which move to adjacent pixels (boustrophedon and Peano) tend to do better than row and Morton

E. DECODING SCAN ORDERS

since Morton and Peano orders are useful but complex, two types of questions arise when they are used:

1. What are the row and column numbers for a given pixel?

2. What is the position in the scan order for a given row and column number?

Method

- start by numbering the rows and columns from 0 up:

  3 10 11 14 15 2 8 9 12 13 1 2 3 6 7 0 0 1 4 5

  0 1 2 3

  - row 2, column 3 is position 13 in the Morton sequence

  1. How to go from row 2, column 3 to Morton sequence? a. convert row and column numbers to binary representations:

  16s 8s 4s 2s 1s 1 0 row 2 1 1 column 3

  b. interleave the bits, alternating row and column bits (called bit interleaving):

  1 1 0 1 row col row col

  c. evaluate this sequence of bits as a binary number:

  Answer: 8 + 4 + 1 = 13

  - so to get the Morton position, interleave the bits of the row and column number

  2. How to find row and column number from Morton position 9? a. convert the position number to a binary number

  16s 8s 4s 2s 1s 1 0 0 1 (8 + 1 = 9) row col row col

  b. separate the bits:

  1 0 row = 2 0 1 col = 1

Generalization

- can express the row and column number to any base, not just base 2 (binary), and including mixtures of bases

- example: row 6, column 15, using base 4 instead of base 2

  64s 16s 4s 1s 1 2 row 6 = 1x4 + 2x1 3 3 col 15 = 3x4 + 3x1 interleaving:

1 3 2 3 1x64 + 3x16 + 2x4 + 3x1 = 123

answer: row 6 column 15 is position 123

- what does this sequence look like? overhead - Base 4 x base 4 scan order
  - arrays of 4 rows by 4 columns, scanned row by row, then repeated at higher levels

- can generate a wide range of possible scan patterns by interleaving digits of different bases
  - the principle of digit interleaving is very widespread, and is built into the PLSS and the GEOLOC grid, as well as numerous systems for map indexing and georeferencing

REFERENCES

Abel, D.J., 1986. "Bit interleaved keys as the basis for spatial access in a front-end spatial database management system," Proceedings, Tesseral Workshop #2, Reading, England.

Franklin, W., 1979. "Evaluation of algorithms to display vector plots on raster devices," Computer Graphics and Image Processing 11:377-397.

Goodchild, M.F., and A.W. Grandfield, 1983. "Optimizing raster storage: an examination of four alternatives," Proceedings, AutoCarto 6, Ottawa, 1:400-7.

Peuquet, D., 1981. "An examination of techniques for reformatting digital cartographic data, Part II, The vector-to-raster Process," Cartographica 18(3):21-33.

DISCUSSION AND EXAM QUESTIONS

1. What systems are used for topographic map indexing in the US and other countries? Discuss the use of digit interleaving in this context, using different national examples.

2. The term metadata is used to refer to information carried with a map layer, such as its accuracy, numbers of rows and columns, type of data stored for each pixel, etc. Discuss the importance of metadata in limiting the operations which a user is allowed to perform on a map layer.

3. Raster and vector have developed as two partially independent traditions in GIS. Summarize the dimensions of the raster-vector debate, particularly in the importance of spatial objects in the two systems.

4. All of the scan orders discussed in this unit visit each pixel exactly once. Discuss the potential advantages, if any, of scan orders which visit certain pixels more than once. Give examples.

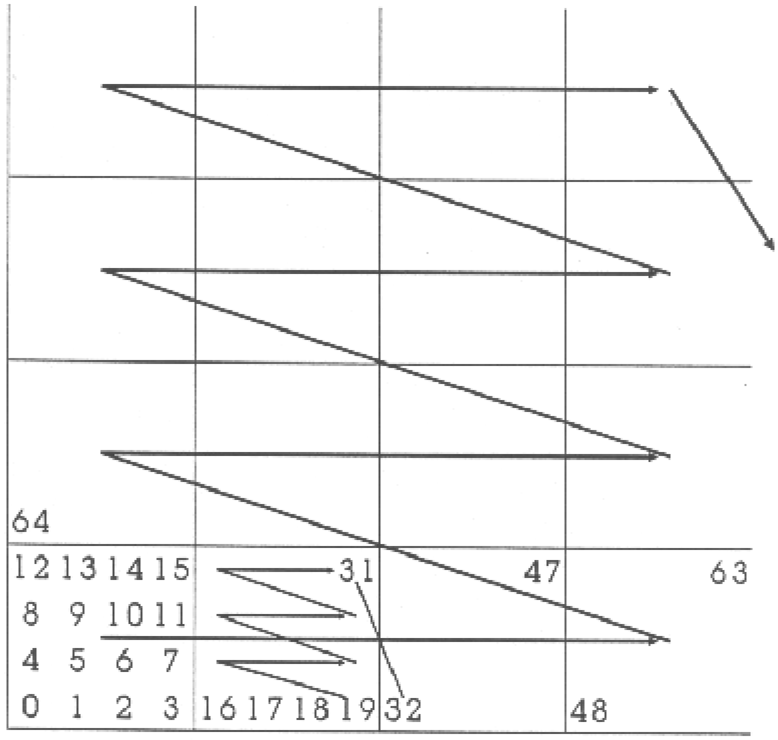
5. Any raster GIS places restrictions on what can be stored in each pixel of a map and what operations can be carried out. Discuss this point as it applies to IDRISI, and any other raster

GIS to which you may have access. Will it let you store an alphabetic value such as A in a pixel and then allow you to carry out arithmetic operations on this layer?

6. Find out what raster storage option (row by row, run encoded, pixel by pixel, layer by layer, etc.) is used by IDRISI and any other raster GIS (GRASS, MAP, etc.) to which you have access.

Last Updated: August 30, 1997.

UNIT 35 IMAGES


64
12 13 14 15
31
47
63
8 9 10 11
4 5 6 7
0 1 2 3
16 17 18 19
32
48