92-07

PROCESSING SATELLITE IMAGERY ON THE INEL CRAY FOR CROP AREA ESTIMATION*

by
Martin Ozga
National Agricultural Statistics Service
United States Department of Agriculture
Fairfax, Virginia

INTRODUCTION

The National Agricultural Statistics Service (NASS) of the United States Department of Agriculture (USDA) has an ongoing program to use Landsat satellite data to augment crop area estimates in certain areas of the USA. Since the areas of interest are rather large, typically major parts of states, a number of scenes are required to obtain full coverage. For example, we are currently working in the Mississippi River Delta region of Arkansas, Louisiana, and Mississippi. A minimum of twelve scenes are required to cover that area. Also, the time between the receipt of imagery and the due dates of estimates can be short, requiring concentrated processing in a relatively short period of time. For these reasons, this project has long relied on supercomputers to do our most large-scale processing, starting with the ILLIAC-IV, an early supercomputer no longer in existance, then using CRAYs at various locations and currently the INEL CRAY.

NASS METHODOLOGY

NASS has for many years produced crop area estimates. These estimates are produced by first dividing up areas into land use strata, where the stratification is based on the percentage of land cultivated. Within each stratum and within each state, a number of areas known as segments are randomly selected. These segments are visited by enumerators who record the crop and field size for the various fields in the segments based both on interviews with the farmers and personal observation. The crop area estimates are generated using statistical techniques similar to those used in opinion polls to extrapolate the total from the sample.

The segment data is a sample of the total land area, while the use of satellite data allows complete coverage of the area of interest. Segments may be located within the satellite image once both the scene and the segment outline are registered to a map base. A digital satellite image, or scene, is composed of pixels. Each pixel represents the reflectance in various spectral bands of an area on the ground. Since the crop types within the fields of each segment are known, this information can be used to determine the relationship between pixel values and crop or other land cover types. The pixel data for known fields of a specific crop or cover type are gathered and clustered to yield statistics, including

---

*Presented at the INEL Computing Symposium, September 21-24, 1992 means and covariance matrices, representing that type. Typically, several output categories represent one crop due to differences in growing conditions. Statistical information from all crops and cover types are put into one statistics file per satellite scene or major analysis district. These statistics files are used to do maximum likelihood classifications of entire scenes. Multitemporal scenes are used wherever possible. These scenes are created by overlaying two different image dates of the same area; usually a late spring and a summer scene. A multitemporal scene can help to distinquish crops which are spectrally similar in a single scene.

Unfortunately, the current satellite imagery is not specialized enough for agriculture and the algorithms for assigning pixels to crops are not well enough developed to obtain a sufficiently accurate estimate directly from the scenes by merely counting pixels. Rather, a regression estimator is used based on the pixel classification into crops and also on the ground data from the segments. Estimation is by land use strata. The strata boundaries are digitized within each county. The digitized counties are registered to maps and thus overlaid on the scenes in much the same way as for segments. Pixels are counted by category and strata. This count is referred to as an aggregation. The aggregation is input to a regression estimator along with the ground data values from the segments to get the final estimation.

LANDSAT THEMATIC MAPPER DATA

The data we have been processing is Landsat Thematic Mapper (TM). A TM scene has 5965 rows and 6967 columns for a total of 41,558,165 pixels. Each pixel represents a square of about 30 meters on each side on the ground and consists of seven channels. Each channel has a value between 0 and 255 so that it occupies one byte. The channels represent the reflectance in particular spectral bands. Thus, the entire scene requires 290,097,085 bytes of data plus some descriptive header information or a little under 300 megabytes. We reformat the data locally from the supplied format to one expected by our programs. This allows us to accomodate new formats or other satellites easily by merely writing a new reformat program. The reformatted data is sent to INEL on tapes, two 6250-bpi tapes per full scene. In our processing, we always use all channels. However, in some cases, the area of interest is only a portion of a scene. In that case, we extract a portion of a scene before doing processing. This extraction is done on the CRAY; the full scene is always sent. As mentioned previously, we sometimes use multitemporal data. A multitemporal scene is twice as large as a single scene so it contains just under 600 megabytes of data.

CONNECTIONS BETWEEN NASS AND INEL COMPUTERS

We use batch processing for all of our CRAY jobs. This is done since our jobs often execute for long periods of time and have complex control language involving execution of several programs and interaction between the CRAY, the CYBER, and the IBM mainframe. Also, batch processing, particularly at lower priorities, is more economical. Since small scale processing is done on our VAX and PCs networked to the VAX, it is necessary to have a connection between the NASS VAX and the INEL CRAY. Unfortunately, we are not on any network also used by INEL. Therefore, the VAX has been set up as an RJE station to INEL using the HASP protocol. Since HASP is a protocol most commonly implemented on IBM mainframes, the RJE connection is really between the NASS VAX and the INEL IBM mainframe with network connections at INEL between the IBM mainframe and the CYBER and the CRAY. HASP assumes three data streams, one for input and two for output. The input data stream is in 80-character records as a holdover from the punched card era. The output streams are the printer stream and the punch stream. The printer stream is used for printed output and thus may contain records of varying sizes. The punch stream was originally intended to punch cards but is now used to transmit binary data in 80-character records. The HASP connection is on ordinary phone lines at 4800 baud using modems at both ends. This connection is used to transmit control language containing small amounts of character data as well as small binary files to INEL as well as to receive printouts and small binary files from INEL. The connection is certainly not fast enough to handle full or partial scenes of either raw or classified satellite data, hence the necessity to mail that data on tapes.

In addition, the path for sending the control language to be submitted to the CRAY is different from the path used to send binary data. The control language goes to the CYBER and is then submitted to the CRAY. The binary data goes directly from the IBM mainframe to the CRAY disk via a network file transfer. Since it is important that no conversion of the binary data takes place, the JCL included with the binary data must be sent in EBCDIC. The binary data is sent to a single file on the CRAY and then broken into individual files by a program run on the CRAY. Hence, for jobs requiring binary input files, two jobs must be submitted, the CRAY job and the IBM job containing the binary data. The job containing the binary data is submitted first in the hope that the binary data will be available on the CRAY disk before it is needed by the CRAY job. If not, the control language is arranged in such a way that the CRAY job will terminate immediately. Similarly, once the CRAY job has completed, output binary data is left on the CRAY. A separate job is then submitted to the IBM mainframe to retrieve this data and send it down the punch stream of HASP to the NASS VAX disk.

The control language for a typical NASS CRAY job is rather complex with several programs being run in one job. Further complexity is added in having to deal with the CYBER and the IBM mainframe. Therefore, a program called CRAY has been developed on the NASS VAX to generate and submit the jobs. This program requires the user to enter the type of job to be performed and the input files required. The CRAY program also assembles the binary input data and creates the control language required to retrieve any output binary data via the punch stream. The latter job must be submitted separately by the user, however, since it cannot be run until the CRAY job has completed.

With the CYBER soon to be discontinued, our procedures will have to be changed to accomodate its replacement. This will mean changes to the CRAY program on the VAX, but should mean minimal and probably no changes in the user interface for submitting jobs.

From time to time, we have considered trying to replace use of HASP with a network connection to INEL. Previously, any serious investigation of a network connection had been delayed due to our long pending move from Washington, DC, to Fairfax, Virginia. Now that the move has taken place, perhaps we can begin to look at the benefits and costs of a network connection.

PROGRAMS ON THE CRAY

All of the programs we use on the CRAY are written in FORTRAN with a diminishing number of assembly language routines. Assembly language was used largely because of certain inadequacies in early versions of the CRAY FORTRAN compiler. These inadequacies have generally been corrected, but we were reluctant to change working code unless it was necessary to make other changes to the programs.

The two jobs which we run on the CRAY are classification and multitemporal file creation. Classification is by far the most time consuming.

The classification is based on a maximum likelihood procedure in which each pixel is independently assigned to a category representing a crop or other ground cover. The algorithm consists of applying a series of discriminant functions to each pixel and assigning the pixel to the class for which the discriminant function yields the highest value. Since the pixels are treated independently, the algorithm vectorizes very well. However, since the algorithm involves matrix multiplies, the CPU time required to do a multitemporal scene is about five times that for a single scene with the same number of categories. The algorithm is linearly proportional to the number of categories. On the INEL CRAY, a multitemporal classification with a large number of categories (on the order of 200) could well take 12 hours of CPU time. Since recent analyses have used considerably more categories than previous analyses done on CRAY systems at other locations, the long CPU time has required us to add a save and restart capability to the classification program. In fact, the classification program was completely rewritten, also eliminating remaining assembly language routines in the process. Fortunately, before the program was rewritten, we never had a situation in which the CRAY went down during a long classification.

Interestingly, there is an optimization to the classification algorithm which may be applied on scalar machines but not on vector or parallel machines. If the classes are arranged in order based on the constant part of the discriminant function, classes with the constant part below the last value computed need not be tested for a pixel. Since, of course, this is not the same for all pixels, such a test would break the vectorization. This optimization saves about 50% of the CPU time on a scalar machine. Nevertheless, the algorithm vectorizes so well that the speedup using a vector machine is considerably greater.

The classify job generally also contains aggregation and sometimes read a window or tape write. Aggregation counts the classified pixels by category and strata, using the overlay of the strata boundaries in the county file onto the scene. Aggregation is not vectorizable, but is done on the CRAY as a convenience since the classified file is available and since both the stratified county file used as input and the aggregated county file are small enough to be transmitted electronically via the HASP connection. When it is not necessary to classify the entire scene, the read window job is used to read the area to be classified, referred to as the window. Again, this job is not vectorizable but is done on the CRAY due to availability of the data. However, both aggregation and read windows run very quickly on the CRAY. Tape write is used to copy the output classified file to tape for later use locally, particularly for displays.

Multitemporal file creation is the other job run on the CRAY. The procedure is to overlay one scene on the other. The multitemporal scene will have the coordinate system of one of the scenes. That scene is known as the primary scene. The other scene is known as the secondary scene. For each primary scene pixel, the closest matching secondary scene pixel is chosen and placed next to the primary scene pixel in the final multitemporal scene. Generating the coordinates of the secondary scene pixel is the purpose of the multitemporal scene registration. Before using the CRAY, a preliminary visual registration of the two scenes is done on the PC. This preliminary registration is used to select a grid of block coordinates for the two scenes, with 64 by 64 blocks for the primary scene and 32 by 32 blocks for the secondary scene. Then, all further processing is done on the CRAY. The blocks are read from the respective scenes, for one channel only and the same channel for both. A simple gradient function is applied to each block to make edges more prominent and decrease seasonal differences. Then, a correlation function is performed between the 32 by 32 block and each 32 by 32 sub-block of the 64 by 64 block. The position yielding the highest correlation value is the shift for that block pair. Taking all blocks and eliminating those with very low correlations or suspicious shifts, least square polynomials are formed describing the row and column of the pixel from the secondary scene needed to match each pixel of the primary scene. These polynomials are applied to each pixel of the primary scene and the appropriate secondary scene pixel is selected and placed next to the primary scene pixel to make the multitemporal scene. The gradient computation, correlation, and computation of the secondary scene row and column are all vectorizable and run well on the CRAY.

PROBLEMS

Generally, use of the INEL CRAY has been quite satisfactory. However, there are a few problems. The most annoying of these problems is the occasional complete disappearance of a batch job. This is due to the batch job containing the statements to copy the output listing to be sent back. If there is some sort of an error which causes that part of the control language to not be executed, no output listing is returned. This is in sharp contrast to batch jobs on mainframes in which a listing is always returned so that errors may be detected and seems to be caused by the batch facilities on the CRAY being less than ideal, particularly in not anticipating that users may want to run jobs through a RJE link.

The other major problem is more local to INEL, namely that not enough disk is available. A multitemporal TM scene requires about 600 megabytes. Another 50 megabytes is required for the output. All of this must be available at the same time. On occasion, that much space is not available in the temporary area, requiring changes to be edited into the machine generated control language when space is found on various disks. With these possible disk problems in mind, we have streamlined our procedures, particularly in deleting unneeded files as soon as possible, but the problem still persists.

A lesser problem is that, since three machines are required for a job, the communication between them can occasionally fail. This problem may not be detected since some of the linkages which we use are not used by very many other users. Generally, the only way to detect such a failure is when no printout is received for a job after a "reasonable" time.