IMAGE ANALYSIS USING ELEMENTAL OPERATIONS AND GENETIC ALGORITHMS

William Stiteler IV
Research Scientist
State University of New York College of Environmental Science and Forestry
Syracuse, NY 13210
wstitel@esf.edu

ABSTRACT

Sources of spatial data for image processing and spatial analysis continue to grow. The challenge when using many spatial modeling techniques is deciding which available pieces of data to use and how they should best fit together. Genetic algorithms (GA) provide a possible method for generating spatial models that use available data in the most efficient method possible. GA attempt to evolve a solution to a given problem by recombining members of a population of candidate solutions over many generations while favoring those found to be most successful. This paper presents a plan for a GA that uses elemental image processing functions as constituents of chromosomes that encode spatial models. A population of candidate models is tested and altered through mutation and recombination until a useful spatial model is found.

INTRODUCTION

Problem Statement

As more imagery and ancillary GIS data has become available, the challenges in remote sensing image analysis have to some extent shifted from finding enough data to use analysis to deciding which data to use and how to apply it. Different applications respond to different portions of the electromagnetic spectrum and to different forms of ancillary geographic data, but exactly which data to use is not always readily apparent. Expert knowledge of either image analysis or specific applications can, of course, help determine which information to use and how it should come together. In many cases, however, it may be that very specific and esoteric combinations of data will help find a solution to a particular problem. Expert knowledge may not help in a case such as this, but there are many ways of deriving these methods by searching for patterns in the data itself. Using training data and an automated technique is a useful way of searching through huge quantities of information in order to use a data set efficiently. It may also remove potential biases that can arise from using only methods that seem to an analyst as though they should work.

Genetic Algorithms

Genetic algorithms (GA) are methods of adaptive computing that search for solutions to a given problem by simulating natural selection. The first step to using a GA is developing a method of encoding candidate solutions as chromosomes. Often, this involves representing a set of parameters to an equation as a string of numbers. For example, a 7x7 convolution filter can be encoded as a string of 49 numbers (figure 1).
An initial population of candidate solutions is created, generally at random, and each chromosome decoded and assessed for the utility of the solution it creates. In the previous example, each number in the string is placed into a corresponding slot in the window, and the window is passed over an image. The resulting output image is then assessed for accuracy versus ground reference. Once fitness is assessed, chromosomes are chosen to be split and recombined with one another, with the more accurate solutions most likely to be chosen. This new generation of solutions is assessed and the process repeats. Ideally, average fitness will improve from one generation to the next. Once fitness ceases to improve (often when all members of the population begin to resemble one another), the process ends and the fittest solution is used. There are many methods of encoding solutions and choosing members for recombination, as well as many refinements to the basic process. In addition to the term chromosome, a number of other terms associated with the field of genetics are used with genetic algorithms (Holland, 1975). Each character in a chromosome is known as a gene, while a gene’s value is an allele. A group of chromosomes is known as a population, and each iteration of a GA is a generation. Members of a population change from one generation to the next both through crossover and mutation. Crossover involves splitting two chromosomes at a randomly chosen gene position and recombining the chromosomes to form new population members. Mutation is the process of occasionally randomly changing single genes. The range of possible values for a given gene, or cardinality, is an important aspect to the construction of a GA. Many GA are constructed with single-bit genes, or genes with a value of either 0 or 1. Assuming the parameters needed have larger ranges, they are decoded from substrings of binary numbers.

The essence of GA, and adaptive computing in general, is the construction of complex behavior or constructs based on very simple rules. This evolved complexity is termed emergence. The fact that simple rules can result in extremely complex output is helpful in finding a method of solving a problem without expert knowledge or potential bias.

**Figure 1.** An example of GA chromosome encoding. A convolution filter is expressed as a chromosome.
USING GA FOR IMAGE ANALYSIS

Image Analysis and GIS Modeling

Genetic algorithms are a flexible technique, and there are many ways of using GA for image analysis. Typically, a GA may be used to evolve a set of parameters for a given equation or technique, used with a particular image or images. For example, Stiteler and Hopkins (2003) used a GA to classify land cover into 6 classes using an unsupervised classification of 64 undetermined classes as input. The GA examined the values in a small area immediately surrounding a pixel of interest to determine the final class of that pixel. The particular pattern of a small neighborhood that resulted in each final class was evolved by the GA. GA have also been used to parameterize neural networks used in image analysis (Russo, 1999; Zhou and Civco, 1996), and for applications such as target acquisition (Smith and Dunay, 1997). If a general model or equation is well understood for a given application, a custom GA can be created to search through many sources of data and parameterize the model.

GA is also very useful for adding ancillary data to remote sensing analysis. A difficulty that often arises when using many spatial layers together is that different layers may have different value ranges, or be of different scales of data. Although there is controversy as to whether the common data scale typology (nominal, ordinal, interval, ratio) is useful for statistical measurement, there is no doubt that certain types of data do not work together with traditional methods of image analysis. For example, image data that has continuous values (ratio data) ranging from 0 to 255 cannot be easily combined with a soils map with a range of 8 possible nominal values. Many methods of image analysis assume the use of normally-distributed ratio data, and in any case varying ranges can have the effect of weighting one data source over another. GA can freely use combinations of various data ranges and scales. This allows for more flexibility in the choice of data sources.

Elemental Operations

While it is not extremely difficult to create a GA for a specific image analysis or GIS modeling application, it can be time consuming. It is also not always readily apparent exactly which operations will be most useful for a given analysis. This leads to the question of whether a GA could be used as easily to create a model framework as it can to parameterize the model. If geographic modeling can be broken into elemental operations, GA can select which of these operations work best to create a specific model, and how they should be combined. A set of elemental operations and their interactions can simply be encoded, and an entire geographic model evolved to suit the requirements of an application. Elemental operations are simply the basic tools from which more complicated analysis tools are created. Some examples of elemental operations are scalar math, overlay, and recoding. Provided enough elemental operations are provided, any sort of more complicated tool can be built from them.

Another advantage of creating a system by evolving analysis tools from elemental operations is that it is more readily customizable to particular applications. Rather than coding a model framework from scratch for each new application, an analyst could use the same basic system for a variety of remote sensing analysis and geographic modeling applications. If the “front end” of the modeling package can be made in a user-friendly form, it also eliminates the need to decide on a chromosome encoding scheme for each application. Once a model is laid out graphically, each variable of the model is encoded as a string of numbers, and the various substrings are combined to form a chromosome.

PROPOSED METHODS

General Modeling Package

The spatially-implicit modeling package for learning and experimentation (SIMPLE) is a geographic modeling package designed to allow a user to graphically place tools and parameterize them to create spatial models. SIMPLE has provision for a variety of GIS and image analysis functions, and allows for iterative procedures. Modifications to SIMPLE will allow its use as a general-purpose adaptive geographic modeling package. The basic format of SIMPLE calls for the user to drag and drop objects representing data and modeling functions and link them to form a graphical representation of a model. The model functions are chosen from a library of possible tools (figure 2).
Figure 2. Data layers and tools are represented graphically to form a model.

These tools range from typical GIS functions such as overlays, and spatial modeling functions such as those used to model the movement of organisms, as well as image analysis functions. Each tool has a set of parameters that are generally set by the user. For example, if the scalar math tool is used, the user must specify which scalar operation is to be used (add, subtract, multiply, etc.), as well as a value to be used. More complicated tools will, of course, require more parameters.

Iterative Problem Solving

SIMPLE handles iterations by providing a graphical construct that specifies which sections of a model will be repeated in a loop (figure 3).
Figure 3. Iterative loops are indicated graphically. The conditions of the loop are specified upon creation.

The loop can repeat for a set number of iterations or until a specified condition is met. In order to allow for the use of a genetic algorithm, several refinements on a basic iteration function are needed. The main difference between a GA and a typical iterative procedure is that a GA must test for fitness at the end of each iteration, or generation. Rather than iterate for a set number of generations, or until a fixed condition is met, a GA will iterate until a test for fitness ceases to show improvement. Another difference between a GA and typical iterative model is that a GA will work on many copies of model, and recombine these models. Typically, the population of a GA will number 100 or more. Testing for fitness can be a challenge for many applications. There are many ways to favor fit members of the population over those that are less fit, but in all cases some test of relative fitness of solutions generated by each population members is needed. Devising a test of fitness can be trivially simple for some applications, but for many real-world problems assessing fitness is more complicated. For many image analysis applications, an overall accuracy assessment of an output image is appropriate, but in other cases a more specialized test may be needed. One potential issue with GA is that because the early generations of chromosomes tend to return poor solutions, fairly detailed sets of reference data are needed to discern subtle differences in fitness (Stiteler and Hopkins, 2002). In order to use a modeling package for GA, a framework to allow for fitness testing must be in place. Existing tools used for the accuracy assessment of spatial data in SIMPLE are used to assess the fitness of model output between each generation. This information is used to favor the selection of certain population members for the next generation.

Parameterization

The first stage of using GA within SIMPLE is to allow for “wildcard” parameters. This will allow a GA to evolve parameters that work best for a particular application. Parameters chosen as wildcards are allowed to vary within a specified range. This replicates a fairly standard use of GA to parameterize a model. In order to take the use of GA further, the spatial modeling tools and input data themselves can be specified as wildcards. In this case, a potential range of tools and input are specified for each section of the model, and the GA is allowed to choose from this set for each member of the model population. For example, a model could be set up to allow for any of three dates of Landsat imagery for a given input data slot, and for any tool from a subset including scalar math, convolution filtering, or overlay for another slot. The parameters for each of these tools are also allowed to vary within a specified range. When the model is initialized, data, tools, and parameters are chosen at random within the specified range for each member of the model population and the model is run. Wildcard data, tools, and parameters can be combined freely with hard-coded portions of the model. Each member of the population is run
and then assessed for fitness, using a user-defined test and criteria (for example an overall accuracy assessment for a land cover classification).

The fact that parameters for each tool must be free to vary along with the tools themselves causes several complications. Because different tools have different numbers of parameters, the length of an encoded chromosome must be allowed to vary. It is also helpful if the user is free to allow the number of tools used to vary, which also requires chromosomes of varying length. Variable length chromosomes are problematic in a GA, because splitting and recombining chromosomes of different lengths will shift gene positions within the chromosome and alter their function when the chromosome is decoded. Another difficulty arising from greater flexibility is the potential for gaps in the useful alphabet of gene encoding. Different tools may need different ranges of values for their parameters, and if the range for a tool that is not used is greater than the range for one that is, it is possible that whole ranges of encoded values will be useless to the model. Ideally, the GA will resolve this by eventually selecting for only those population members that do not have useless gene values, but having significant numbers of poorly-performing solutions degrades the efficiency of the GA. A method for ameliorating this problem is using single bit encoding. In some cases, it may be more natural to use alphabets of larger cardinality. For example, if all of the parameters for a given problem range from 0 to 10, it may make sense to use genes that each range from 0 to 10. However, it has been shown that any GA can be encoded with single-bit cardinality, and this prevents potentially worthless gene values from entering the population. It also allows for easier use of a mutation operation (single gene values are simply flipped), and allows for smaller populations that nonetheless have representatives of every allele (Golberg, 1989). Single-bit encoding does increase the length of chromosomes, because, for example, it takes 8 genes to express a range of numbers from 0 to 255.

CONCLUSIONS

Genetic algorithms have great potential for image analysis and GIS modeling. GA can perform a massively parallel search through input data to find a set of parameters that best work with a given equation or model. It has similar potential to search through a set of possible analysis tools or elemental components to tools to create a geographic model. Very simple analysis tools can be recombined and parameterized to form very complex tools and models. The challenge is finding which combination of simple tools creates the necessary complex tool or tools. The use of genetic algorithms is an ideal method of using simple components and behavior to evolve much more complex constructs.

REFERENCES


