THE USE OF EVOLUTIONARY AND ADAPTIVE COMPUTING METHODS FOR REMOTE SENSING

William Stiteler IV
Dr. Paul Hopkins
State University of New York
College of Environmental Science and Forestry
Environmental Resources and Forest Engineering
312 Bray Hall
Syracuse, New York 13210
wmstitel@syr.edu
phopkins@syr.edu

ABSTRACT

The field of evolutionary computing includes many concepts with the potential to advance knowledge in the study of remote sensing. Evolutionary and adaptive computing is a bottom-up process that involves moving toward a solution based on simple rules and external feedback that often mimics biological mechanisms. Automating the process of developing parameters for complex operations has several advantages. Genetic algorithms may be used to evolve solutions to problems or sets of parameters where optimal solutions are not thoroughly understood, or where expertise is not available. They can also reduce the amount of time an expert operator must spend analyzing different potential solutions to a problem. Often in remote sensing applications, a number of potential solutions to a given problem exist, with no clear best solution. Autonomous agent methods are systems that develop behavior and derive a solution based upon the environment. These methods can be used similarly to classification methods commonly used in remote sensing work, and for other spatial analysis. Cellular automata is another branch of evolutionary computing that is commonly used in spatially-oriented activities such as forest fire modeling and has potential for other branches of GIS and remote sensing. Two-dimensional cellular automata are organized as a grid of cells, and therefore can be easily adapted for use with raster images. Adaptive computing methods are already being used in remote sensing and GIS analysis, but there are many more potential uses of the technology. This paper will introduce several evolutionary and adaptive computing methods and examine how they might be applied to the field of remote sensing and GIS analysis.

INTRODUCTION AND BACKGROUND

There are many broad and overlapping terms for methods referred to as evolutionary computing, such as complexity and artificial life research. Artificial life and adaptive computing are being used for an increasing number applications in various fields. Artificial life is a broad field, comprising such topics as genetic algorithms (GA), autonomous agents, and cellular automata (CA). Some applications of adaptive computing include optimizing traffic flow patterns, modeling forest fires, and developing parameters for neural networks.

Adaptive computing methods excel at finding potential solutions for problems that have no one, clear "best" answer, and wide ranges of parameters. Many problems related to remote sensing analysis fit this description. Digital remotely sensed imagery is broken into discrete pixels, each with a single value per band, but it represents values which are continuous both spectrally and spatially. Often, even direct ground observation cannot delineate specific areas where one and only one class of interest ends and another begins. One example would be an area that contains a forested wetland that gradually becomes dryer forested land. If a wetland class and a forest class were both of interest, it could be difficult to decide upon a single correct value for every pixel in the area. This is particularly true given that slight shifts in an image location could place a given area within different pixels, and subtle changes may occur within a single pixel. Another example would be the definition of land use. Single areas in an image could potentially belong to several classes simultaneously, and changes in the arbitrary definition of classes could result in very different final products.

In both of the examples above, analysis based on remote sensing analysis could not return a single, definitive answer for all areas and for all applications. A method that returned different answers in different runs, even with the same parameters, might therefore be reasonable. Many artificial life methods respond with this "black box" approach. As with natural life, it is often not readily apparent why an artificial organism acts in a particular way. If properly done these methods will eventually arrive at a useful solution, although possibly in an unorthodox manner.

Adaptive computing is also, as the name suggests, useful for applications within changing conditions. In the example of the wetland above, a given area might be appear to be more or less like a wetland during different times of the year, or at a different ground sample resolution. A method that could adapt to different input conditions could therefore be useful. Adaptive computing is also useful for modeling in dynamic environments. For example, fire modeling benefits from methods that can adapt to changing weather conditions, and inherently chaotic inputs.

A defining characteristic of adaptive computing and other methods that fall under the broad category of complexity analysis is complex behavior driven by simple rules. Behavior that arises in an unpredictable fashion given simple inputs and rules is termed "emergent". The classic example of emergent behavior is that of ant colonies (Dorigo and Colorni, 1996). Colonies of ants exhibit very complex behavior, driven by interactions between individual ants. The behavior of ant colonies is not directed by any guiding intelligence, but rather by individual ants that each perform based on simple rules.

There are many areas of adaptive programming that have potential for use with remote sensing research. The following sections each describe one of these areas and provides a general overview of the types of remote sensing applications most appropriate for each.

GENETIC ALGORITHMS

Introduction

Genetic algorithms search for a solution to a problem by simulating natural selection. Potential solutions are encoded as a string of numbers, and an initially random population of solutions is tested for fitness. Potential solutions are then mated in pairs, resulting in new solutions that are combinations of the original ones. Pairs for mating are selected to favor the fittest members of the population. This process continues through many generations until a useful solution is found or the population as a whole ceases to improve. In effect, a GA works by instituting a massively parallel search for a solution. Information on how well a potential solution or piece of a solution is working is implicitly shared throughout the population.

A simple example of a GA is the parameterization of the formula $3x - \frac{1}{2} y = z$. If various values of z can be assigned relative fitness in some way, a GA can be constructed to find a good answer by expressing x and y in a numerical string. If the alphabet used to construct the chromosome is limited to 0 or 1, it will likely take several genes to express each parameter. For example, a chromosome used to express values for x and y that can range from 0 to 255 might look like 0110001010010011, with the first eight characters representing x and the next eight representing y. The value of z created by this chromosome would be $3*98 - \frac{1}{2}* 147$, or 220.5.

Determining the relative fitness of chromosomes is sometimes trivially easy, as in the case of a GA that performs a land cover classification. In this case, accuracy can be assessed for the classified image and overall accuracy or some other accuracy statistic used to represent fitness. In other cases, it may be more difficult to arrive at a single number to represent fitness, or this number may only be available after a number of post-processing steps. For example, a GA that is used to develop a convolution filter does not yield an immediate fitness value. Only after the filter had been applied to an image and accuracy on that image assessed would the usefulness of the filter be apparent.

Once all members of the population had been assessed for relative fitness, pairs would be selected for recombination. One common way to select population members is to create a "roulette wheel" upon which each member of the population is given a number of slots proportionate to its fitness value. The wheel is then spun twice to select members to mate. In the example above, the chromosome 0110001010010011 might be mated with 1100010100101001. A randomly-chosen crossover point could produce the following two offspring:

0110001010010011 → 0110010100101001
1100010100101001 → 1100001010010011

Members of the new population produced in this manner would be assessed for fitness and the process repeated. Other methods of choosing the fittest members of a population, such as methods that select based on fitness rank rather than actual fitness value. There are also various methods of choosing crossover. For example, several crossover points may be selected rather than only one, and at times population members may not be recombined at all, instead transferring unchanged to the next generation. Mutation, in which a single gene of a chromosome is occasionally changed, is another common GA operation. This can serve to spark change in an otherwise stagnating population and help ensure that potentially useful adaptations are not permanently lost.

After several generations across an entire population of candidate solutions, a GA will ideally begin to converge upon a good solution. Further modifications to a basic GA can introduce "species" that will specialize at looking for different solutions to a single problem. Mechanisms to simulate predation or parasitism can force a GA off a moderately good solution to look for better solutions, and the use of diploid chromosomes allows recessive traits to emerge when needed.

Remote Sensing Applications

GAs are general-purpose tools that can be used to search for solutions to almost any problem, although they are more appropriate for some applications than others. Any application of remote sensing that requires parameterization could benefit from the use of GAs. A basic supervised or unsupervised classification could use a GA to develop parameters such as which layers are used for a classification and which rules are used to perform the classification. The various parameters would be represented as a numeric string, with each parameter represented by a gene or subset of several genes. A classification would be performed with each of a population of randomly generated parameters and fitness assessed through standard accuracy assessment methods. The best-performing sets of parameters would be split up and recombined with each other, and the process repeated until no further improvement resulted. A more complex version of this arrangement could allow a piece of a chromosome to represent post-processing, such as the application of low-pass filters of varying sizes. A GA could also be used to perform a classification from start to finish, rather than simply finding the parameters for another classification method. In this modification of a supervised classification, the spectral response of various bands and a final output class would all be encoded as a chromosome. The various combinations of input and resulting output would all be compared against accuracy reference over multiple generations to find the most fit chromosomes.

GAs could also be useful in finding useful ways to use multiple sources or types of data. It can be difficult to use ancillary data or fuse two different types of imagery because these types of imagery may have different ranges. A particular difference in value may mean something completely different in one band of an image than it does in a digital elevation model, for instance. A GA evolves a solution without making any assumptions about its various inputs, and can therefore be used to bring together any number of different types of data.

AUTONOMOUS AGENTS AND CLASSIFIER SYSTEMS

Introduction

Autonomous agents are programs capable of independent action and adaptation in changing environments. In some ways, they are similar to expert systems, in that they are used to replace human intervention in solving some problem. However, expert systems tend to be set up in a top-down fashion, where the response of each particular situation is explicitly programmed to simulate a human expert. Autonomous agents are programmed bottom-up. That is, they start with a few simple rules and evolve appropriate responses to any situation that arises.

The bottom-up characteristics of autonomous agents give them their flexibility. If inputs change, autonomous agent can adapt more readily than an artificial intelligence method that is rigidly programmed (Harp and Samad, 2000). This flexibility can also be useful for situations in which an appropriate response or method is not well understood, even if input conditions remain static. Different methods are tried until those that return good results are found. This removes the necessity of knowing the best path to a solution ahead of time.

One variant of the autonomous agent is the classifier system. A classifier system as defined in the field of artificial life has little to do with "classification" as the term is generally used in remote sensing research. A classifier system is an entity that senses its environment and reacts to changes in the environment through a list of messages and a set of rules (Goldberg, 1989). Each message has an input component and an action component, and the population of messages evolves through the use of a GA. Any messages that match the current conditions may bid to be used. The currency used to make these bids is won through rewards based on useful behavior. Rewards are apportioned throughout the immediate message that created the behavior and the messages that led to the current message. This allows for an agent that can act in ways that have no immediate payoff, but which set up future useful actions. Actions are useful, but are triggered based on conditions unlikely to exist initially may be set up by actions without immediate payoff, and both sets of actions receive a reward.

Remote Sensing Applications

Top-down artificial intelligence methods such as neural networks and expert systems have already found use in many areas of remote sensing (Frizzelle and A. Moody, 2001; Ji, 2000; Foschi and Smith, 1997; Huang and Jensen, 1997). Autonomous agents may be used in a fashion similar to these methods. An autonomous agent can determine how to merge ancillary data with remotely sensed imagery, or which final land cover class should result from a particular spectral response pattern. Essentially, any part of process that requires human input has the potential to be replaced with an autonomous agent, much as it could be replaced by an expert system that is programmed by a human expert.

Autonomous agents can also be used to simulate the behavior of people or animals in a GIS model. A map can be populated with artificial organisms driven by simple rules and the ability to dynamically alter their behavior. An example is habitat suitability modeling. Agent technology can also be used in a more abstract fashion to simulate the behavior of large groups of individual entities.

A classifier system, in particular, is useful for applications that require multiple steps. In many applications, there may be situations in which a given set of inputs does not immediately suggest a single course of action. An example of this is a process with multiple inputs or potential inputs as well as various potential post-processing methods, such as a land cover classification procedure with multiple steps. A particular filtering procedure may improve the classification accuracy, but only when based on an image that results from another filter which does nothing itself to increase accuracy.

CELLULAR AUTOMATA

Introduction

Cellular automata are constructs that have a set of possible states, and are governed by a few simple rules. A two-dimensional CA is organized on a grid, much like a digital image. The rule that determines the state of a given cell of a CA at time t+1 uses its state and the state of its neighbors at time t. The simple rules that a CA uses can result in very complex behavior. The most famous CA is Conway's game of life. The Game of Life is a two-state CA with the following rules: A cell with state 0 and exactly three neighbors changes to state 1. A cell with state 1 and 3 or 4 neighbors remains at state 1. All other cases result in a state of 0. Despite having only three rules and two possible states, Life can show very complex behavior. Patterns can be made to exist that remain unchanged over time, or exhibit periodic behavior. Other patterns will spawn new patterns or move across the grid. Figure 1 shows a starting condition known as R-pentomino. This starting pattern does not stabilize until generation 1103. At generation 1103 the CA consists of 25 stable objects that fit into a 51 X 109 pixel rectangle. Many of these 25 objects will continue moving off in different directions, but they no longer interact with each other. The potential for this complex configuration of objects is contained within this starting pattern and the 3 rules that govern Life. Life has even been shown to be a universal computer (Berlekamp, *et al*, 1982). A large enough CA using the rules of the Game of Life can perform any task that can be programmed on a computer.

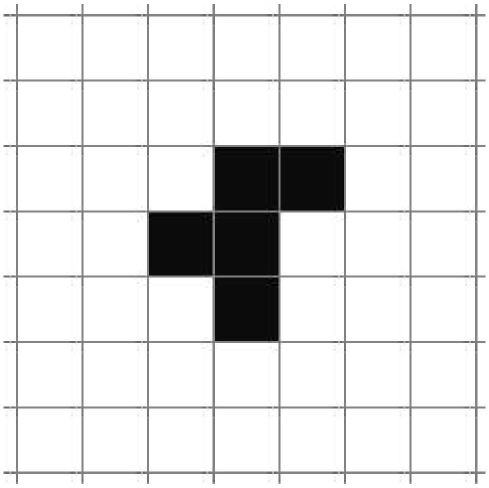

Figure 1. The R-pentomino pattern.

Although a one-dimensional CA can perform any task that a two-dimensional one can, two-dimensional CA lend themselves to use with digital imagery. Among the parameters that may be varied within a CA are the size of the neighborhood that is taken into account for rules and the number of possible states a cell can have. As these parameters expand, it becomes easier to model more and more complex behavior, although at the same time it becomes more difficult to predict and control what a CA might do.

Remote Sensing Applications

Two-dimensional CA are often used in GIS modeling. The way in which a CA acts on both spatial and temporal variation makes it ideal for use in modeling phenomenon such as the spread of fire or the growth of urban areas. Many of these applications emphasize change over time on a landscape. Remotely sensed images are valuable for these applications, if only as an initial input (Li and Yeh, 2001).

A key aspect of CA is the way that they examine local neighborhood operations. Changes are made to a single CA cell based on the immediate neighborhood of the cell, so a CA can be used to examine phenomena such as spatial correlation that are often seen in land cover and land use. The neighborhood examined by the CA can be varied, from those cells directly in contact with the cell in question to larger neighborhoods that span many cells. These neighborhoods need not be square, although they typically are.

CA can also be used in a fashion similar to filters often used in remote sensing work. A CA stores the state of an entire grid at time t before changing the states, and therefore has an effect similar to a filter that runs over an image and outputs a changed image. The main difference between filters and CAs is that while a typical filter will perform a single mathematical operation using values in a neighborhood to find a new value for one cell, a CA will change the cell value based on one of a set of several rules. A CA has the potential to perform sophisticated filtering because of the very complex behavior it can perform.

DISCUSSION

Appropriate Applications

Adaptive computing is more appropriate for some types of applications than others. In general, adaptive computing methods work well when there is a degree of uncertainty not only in the path to a solution but in the solution itself. Applications that use or produce things that are difficult to pigeon-hole may be appropriate for adaptive computing methods. Even if input or output of continuous phenomena is broken down into discrete categories, adaptive computing may be useful. The dynamic nature of these methods allow for the fact that an input could belong to more than one class, or fall between classes. This flexibility also means that a given set of inputs can produce one of several answers. As unsettling as this may seem, it often reflects reality. For example, digital images are an attempt to break two continuous phenomena down into discrete chunks: the electromagnetic spectrum and the spatial landscape. The spectral response of a given area of the ground is not only continuous, it is subject to significant change with changing conditions. All of the ground within a given pixel (and some of the surrounding area) contributes to the spectral response of that pixel. A change of pixel resolution or a shift in pixel boundaries will result in a significantly different image. It makes sense that for a method dealing with such input to allow for a certain amount of variation. Applications such as land cover classification that attempt to define hard classes for an inherently fuzzy reality can also benefit from methods that at least allow for differing interpretations.

Some types of applications simply do not respond well to adaptive computing methods. Some problems have inputs that fall into more rigid categories, or solutions that can be easily enumerated. Such applications generally respond better to other methods. For example, the problem of finding a set of parameters for a mathematical formula that will return the highest possible value can certainly be solved using a genetic algorithm, but it may be more efficiently found through other methods.

Combining Methods

Adaptive computing methods have a great deal of potential for use with remote sensing applications, and are often even more powerful when combined. The common characteristics of the otherwise different methods often make them complementary. GAs, perhaps the most general-purpose of the methods discussed, are often used to find parameters for other methods. For example, as discussed previously, GAs are an integral part of classifier systems.

There is a natural division in methods of working on remotely sensed data between those that act directly on the raw data of imagery and those that work on the interaction of bands of images or whole images. One scale is concerned with individual pixels or groups of pixels, while the other is concerned with how whole images interact. Methods at each of these scales of focus may work in a complementary fashion. For example, CA, which by their nature are most appropriate for working directly on the cells of a particular image, may work well with methods that work on a more macroscopic scale.

Computational Demands

Adaptive computing methods are often computationally intensive. For example, a typical GA might have a population of 100 members and run for 50 or more generations. Fitness must be assessed and matches made for each of the members of the population for each generation. By their nature, adaptive computing methods tend to explore a wide range of the solution space for a given problem. This can result in solutions that more tightly-focused methods would not consider, but it may require more computing power.

SUMMARY

Adaptive computing and artificial life methods have great potential in the field of remote sensing, as they do in many other fields. The methods discussed in this paper are generally quite flexible. Some applications better than others to adaptive computing techniques, however. In general, applications that have multiple or poorly-understood solutions are better suited for use with adaptive computing than problems with only one correct solution. Many, if not most, remote sensing applications meet these criteria because they do not have single, obvious solutions.

REFERENCES

Berlekamp, E. R., J. H. Conway, R. K. Guy (1982). *Winning Ways, for Your Mathematical Plays.* Academic Press: New York.

Dorigo, M., and A Colorni (1996). The ant system: optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics-Part B, 26(1):1-13.

Foschi, P. G. and D K. Smith (1997). Detecting subpixel woody vegetation in digital imagery using two artificial intelligence approaches. *Photogrammetric Engineering and Remote Sensing*. 63(5): 493-500

Frizzelle, B. G. and A. Moody (2001). Mapping continuous distributions of land cover: a comparison of maximum-likelihood estimation and artificial neural networks. *Photogrammetric Engineering and Remote Sensing*. 67(9): 693-705.

Golberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley: Reading.

Harp, S.A., and T. Samad (2000). Adaptive agents and artificial life: insights for the power industry. In *Soft Computing and Intelligent Systems,* N. K. Sinha and M. M. Gupta, eds. Academic Press: San Diego, pp. 571-592.

Huang, X. and J. R. Jensen (1997). A macjine-learning approach to automated knowledge-base building for remote sensing image analysis with GIS data. *Photogrammetric Engineering and Remote Sensing.* 63(10): 1185-1194.

Ji, C. Y. (2000). Land-use classification of remotely sensed data using Kohonen self-organizing feature map neural networks. *Photogrammetric Engineering and Remote Sensing*. 66(12): 1451-1460.

Li, X. and A.G. Yeh. (2001). Zoning land for agricultural protection by the integration of remote sensing, GIS, and cellular automata. 67(4): 471-477.

Skidmore, A. K., F. Watford, P. Luckananurug, and P.J. Ryan (1996). An operational GIS expert system for mapping forest soils. *Photogrammetric Engineering and Remote Sensing*. 62(5): 501-512