APPLICATION OF HEURISTIC OPTIMIZATION TECHNIQUES IN LAND EVALUATION

A thesis presented to

the faculty of

the College of Art and Sciences

of

Ohio University

In partial fulfillment

of the requirement for the degree

Master of Science

Valeriy Kovalskyy

August 2004
This thesis entitled

APPLICATION OF HEURISTIC OPTIMIZATION TECHNIQUES IN LAND EVALUATION

By

Valeriy Kovalskyy

has been approved for

the Program of Environmental Studies

and the College of Art and Sciences by

James K. Lein

Professor of Geography

Leslie A. Flemming

Dean, College of Art and Science
The research constitutes the attempt to create new approach to optimization in the field of land use planning. It combines methodologies of remote sensing and landscape ecology, bringing together multi-spectral analysis of digital imagery and analysis of landscape texture. These powerful tools are used to classify and cluster the area of study to the best advantage that can be predicted in developed model. This means that the developed procedure can help to configure or redistribute the area and resources among land use types in a manner that allows maximization of output, which can be received from utilization of the resources. In contrast to the traditional land use assessment and optimization techniques used by USDA and FAO, this methodology does not use linear optimization for individual map unit. When running the optimization, developed model uses the idea of common effort and possibility to bring together all the necessary resources from different map units that can help to achieve the goal of a particular land utilization type. Based on those ideas, the algorithms of semi-lacunarity analysis and edge search were created and combined into one procedure of raster based heuristic land use optimization. Also, the structure of participating data types were designed for the need of proper input data storage and manipulations. The procedure was tested on the soil and terrain data obtained in Wayne National Forest (Ohio). The map of Optimized Land Use became the result of the research and testing. The model helped to exclude ineffective
land uses and reassess the land to the effective ones, while keeping their distribution reasonably close to natural patterns of resource distribution.

Approved:

James K. Lein

Professor of Geography
# Table of Content

Abstract .................................................................................................................. 3

List of Figures ......................................................................................................... 7

List of Tables ........................................................................................................... 8

Introduction ............................................................................................................ 9

Chapter 1. Theoretical Background of the Heuristic Land Use Optimization .... 13

1.1. Developing The Model .................................................................................. 13

1.2. Heuristic Optimization versus Linear Optimization .................................. 18


Chapter 2. Methodology ....................................................................................... 27

2.1. Implementing the Model .............................................................................. 27

   2.1.1. The Structure of Objects in the Procedure ....................................... 31

   2.1.2. The Algorithm: Procedures ................................................................. 32

2.2. Analysis of Algorithm Performance ......................................................... 34

   2.2.1. Time Complexity ................................................................................. 34

   2.2.2. Space Complexity ............................................................................. 37

2.3. Strategies and Options in Performing Analysis ......................................... 38
Chapter 3. Testing and Results ................................................................. 42

3.1 Study Area ....................................................................................... 42

3.1.1. Input Data Preparation ............................................................. 47

3.1.2. Deriving Signatures ................................................................. 49

3.2. Flow of the Analysis ..................................................................... 52

Conclusions and Discussion ................................................................. 59

Bibliography ......................................................................................... 64

Appendix A: C++ Code for the Model and Procedures ......................... 67
List of Figures

Figure 1.1. Resource Availability distribution according to the FAO Methodology. ….. 13

Figure 1.2. Differentiated (natural) Resource Availability distribution………………. 14

Figure 1.3. Consideration of Resource Distribution in the Model (Ra - resource availability). …………………………………………………………… 15

Figure 1.4. Assigning land uses (P - performance). …………………………………… 16

Figure 1.5. Relative Performance on Integrated Area (P - performance). …………….. 17

Figure 1.6. Extensive Side of the Model. ……………………………………………… 18

Figure 1.7. Overcoming Pure Pixel Problem. …………………………………………… 21

Figure 1.8. Signatures and Factual Data in the Model. ……………………………….. 23

Figure 2.1. Differences in Strategies 1 and 2. ………………………………………… 40

Figure 3.1. Location of the Study Area. ………………………………………………… 42

Figure 3.2. Distribution of Soil Types on the Study Area. …………………………… 45

Figure 3.3. Terrain Characteristics of the Area: (a) Elevation; (b) Slope. …………… 46

Figure 3.4. Grid of Points For Interpolation. …………………………………………… 47

Figure 3.5.1 Input data: a) – Slope; b) – Depth of Soil. ……………………………… 48

Figure 3.5.2 Input data: a) – Water availability; b) – Organic Metter. ………………. 49

Figure 3.6. Distribution of Forest Types within the Study Area According to Strategy 1. …………………………………………………………………… 53

Figure 3.7. Distribution of Forest Types within the Study Area According to Strategy 2. …………………………………………………………………… 54

Figure 3.8. Optimized Distribution of Forest Types. ……………………………….. 56
List of Tables

Table 1. Physical Properties of Soils Found within the Study Area. ..................43

Table 2.1. Signatures for Forest Types of Condition Class 1-3 and Age more than 70. . 50

Table 2.2. Signatures for Forest Types of Condition Class 6-4 and no Age Limitation. . 51

Table 3. Differences in Distribution of Forest Types between Strategies 1 and 2. .......... 55

Table 4. Differences between Optimized & Un-Optimized Distributions of Forest Types. ................................................................. 57
Introduction.

Land Use has always been the question where two factors – nature and people – were coming together for purpose of continuation of each other’s future. Pessimistic predictions that have appeared since first publication of Rome Club Report, promise a quick reduction of land resources within modern level of consumption. Being developed in one direction, some regions have almost run out of their resources. Therefore, quickly growing in the past, regions are stagnating and undergoing underemployment, low wage rates and increase of poverty. These events are accompanied with environmental problems as a results of past intensive exploitation of the resources. Such events make people look for other opportunities of land use and find new directions of the development.

One of the tools for determination of future directions of the development is optimization of land use via economic evaluation of land (EEL) for different purposes or diversification of the land use. This tool is not only the simple measuring of economic interest or money that might be earned from land, but also the determining of possible duration of certain land use that must support the original intent of the individual land users to optimize the use of their land. Geologists, when counting the capacities of useful element in rock for mining, did first attempts of economic evaluation of land. Only one land characteristic was considered, and the question that EEL responded was whether or not this land fits for industrial development. Such an approach to land evaluation is recognized as simple evaluation for one purpose.
The development of many regions presents facts that it is not necessary for economic growth to be related only to one type of production. Generally, economic development of a region is built on few productions that can make goods with competitive price and quality. Therefore, the simple evaluation cannot give a complete description of the land usage potential. This idea has led to the appearance of complex economic evaluation of the land that consists of several simple EEL and can give information about possible combinations of land development. Within the complex evaluation, the economic development of region might have a better perspective for the economic growth.

Pioneering complex land evaluation method was created by the FAO for evaluation of agricultural potential for LDC (Less Developed Countries). This evaluation presents the expected productivity of several crops that depends on a set of soil characteristics. The methodology has been used in many UNDP projects, and by many national agencies in developing countries, with modifications and simplifications made to adopt it to local conditions. The framework of this methodology is “practically unknown and non-influential in the USA” (D., G., Rossiter, 1996), since the country has its own Land Evaluation and Site Assessment system, designed in 1981 by the Natural Resource Conservation Service. One of the most unique characteristics of LESA was that it “provided a national model with consistent terminology and a set of classification procedures using soil-based and other site factors while offering a great deal of local flexibility” (James R. Pease and Robert E. Coughlin, 1995).
FAO methodology defines EEL as “The process of predicting the use potential of land on the basis of its attributes” in its Framework and Guidelines for land evaluation. Performing the evaluation - factual land characteristics should be summarized as a set of conditions present on the area or area resource base R. This set is then compared with Land Use Requirements (LUR) of particular Land Utilization Type (LUT) and the result of this comparison determines the level of land severity or degree of expression of a Land Quality (LQ). Additionally, overall characteristic of the land is included in this evaluation, and called Land Sustainability, which is “the fitness of a given Land Mapping Unit (LMU) for land utilization type” (Rossiter, D.G, 1996). This complex evaluation brought a significant progress in development of agricultural sector in Developing Countries.

The FAO’s land evaluation framework helped to provide a useful starting point for EEL, being based on many years of expert consultation. However, these methods involve many repetitive calculations or table lookups if many alternatives are to be compared. Manual procedures, both for construction of matching tables or similar methods, and for calculation of suitability, were time-consuming and error prone (D., G., Rossiter, 2001). Using this approach, FAO specialists helped to determine the priorities in land use, while problem of proper placement still depended on the map unit defined in the survey. This means that evaluation was done on the generalized vector defined parcels, and based on sustainability indices (Tucker C. C., & Irwin U. A., 2002). Consequently, the choice was made without taking into account surrounding areas. Therefore, FAO’s EEL was choice-per-unit oriented procedure, but not shape plus choice oriented optimization.
The purpose of this study is to free the land use of previously defined (human biased) boundaries and let it determine its optimal shape, while using the benefits of FAO methodology, plus take into account surrounding conditions. This may bring the land evaluation and optimization closer to the natural patterns of resources’ distribution and therefore help to utilize these resources more efficiently.
Chapter 1. Theoretical Background of the Heuristic Land Use Optimization

1.1. Developing The Model

One of the USA’s most recognized specialists in land evaluation - David G. Rosier - defines EEL as a procedure “leading to rational land use planning and appropriate and sustainable use of natural and human resources”. Land evaluation and optimization had their origin in land capability classification, in which the potential of land utilization is expressed in terms of its predicted physical response to various land uses or in terms of physical constraints to these uses (D. G. Rossiter 1994). The key factors in this evaluation are the resources and more precisely it is their ability to be converted into some type of utility and provide support or continuation for this utilization. According to the FAO methodology that cannot go beyond the predefined map units, the availability of resources within one map unit is generalized to one level and their distribution can be described as in Figure 1.

![Figure 1. Resource Availability distribution according to the FAO Methodology.](image-url)
The methodology then can compare resource availability (sustainability levels) of different land use types and choose one with the highest value. However the real picture in nature would look more like in Figure 2.

![Figure 1.2. Differentiated (natural) Resource Availability distribution.](image)

The ability to differentiate or see the space – value change plays significant role in defining the shape of future land use unit. It means that the generalization is still a necessity as the choice of land use depends not only on resource availability but also on its feasibility level (if there are not enough resources, there is no point to develop the land use even if it has the highest level of resources on the area). Therefore, the land use must have the feasibility constraint while searching for its place on the map. In the same time, high differentiation level in resource distribution helps the land use to absorb all the area that may contribute in accomplishing the function of this land use, while preserving the unity of its generalized body. In other words, the optimization, here, should provide not
only the right choice of land use for the area but also right distribution of land uses that would keep the overall picture of land use performances in the frames of feasibility.

Following these criteria, the model for land use optimization should have a consideration of resource distribution different from one used in FAO methodology. Therefore, this model should view this distribution as continuous data of absolute resource values, but the land use should evaluate the resource combinations and amount for itself.

Figure 1.3. Consideration of Resource Distribution in the Model (Ra - resource availability).
While making the choice for one land use over another - assuming that the performance of land use is in direct dependence with resource availability, the model should take into account feasibility level and assign the land uses as follows:

![Diagram](image)

**Figure 1.4. Assigning land uses (P - performance).**

On infinitely small portions of the area, the absolute performance would vary as shown in Figure 4. However the task of optimization is not only to maximize performance on every individual portion but also to integrate the land uses into one system that utilizes the resources efficiently on the whole extent as well. Hence, the performance should be considered as integrating factor in optimization, as factor that connects individual portions into groups forming shapes (patches/clusters) of the land use. Within this context the relative performance indicator should be used, such as mean performance of the whole land use patch per unit of area.
The Figure 5 shows that the overall performance can integrate the area portions into one land use patch that still keeps the performance level in limits of feasibility (the portions with level higher than effective would compensate for ineffective portions). In the same time, the multiple (combined) lad use optimization eliminates areas of ineffective use replacing them with other effective land uses and brings the overall performance level higher then feasibility level. This aspect of the model represents the intensive side of the optimization.

Let’s consider the case where some areas that have insufficient amount of resources cannot be replaced with other high performance land uses. Leaving this area and its resources without utilization would lead to inefficient land use. Therefore, the model should also connect those areas with land uses to the degree that would not threaten the overall performance.
In this case the optimization helps the land uses to absorb the individually ineffective portions of area lowering the overall performance level but increasing the volume of output (relative performance * area).

By this generalization the model does not say that there will be evenly distributed performance throughout the extent (there will be still high and low performance areas), but it does say that it will choose and place the land uses in a manner that would provide feasible level of outputs maximally adopted to natural patterns of resource distribution.

1.2. Heuristic Optimization versus Linear Optimization

FAO methodology uses optimization to choose land use for an area based on its defined characteristics. The objective function of this optimization is to maximize the Sustainability Indices or performance of the area (D. G. Rossiter 1994). The constraints are denoted as various manipulations with input data from maps and tables which therefore considered as constants. This allows variables and equations to be independent.
from each other, so that linear optimization and simplex method, in particular, can be used to make decisions in this case.

In case of the model presented in this study, the variables of performance are in strict dependence on area of land use patch that also varies (mean performance = \( \frac{\sum \text{individual portion performances}}{\text{area}} \)). Because of this dependence the constraints and objective function cannot be denoted as linear equations, and thus this model cannot be considered for linear optimization. Also the model is just a planning tool and so far it cannot provide the evidence or hard proof for its effectiveness, though it makes sense to an analyst to use it in her research and therefore the model can be classified as a heuristic optimization model.


When the benefits of presented approach are clear, the problem would be just the means, which can help to accomplish the task. The methods of field research would be suitable but time and cost consuming ones. Even though there are many results of studies available to analysts, not all of these are organized with same level of detail and measurement units that would probably require more effort to bring to the common base. The research must be build on some efficient methods promising to take into account a broad extent of research area and high level of detail measurements with common base of raw data that can provide a simple transformation into numeric values and further multipurpose interpretation. With consideration of all listed above factors the use of technology becomes reasonable and necessary mean of land quality evaluation.
Technology enhanced the process of land evaluation significantly, especially use of GIS and Digital Remote Sensing Imagery. Satellite imagery brought the convenience of synoptic view on the Earth that can help to receive the information about various characteristics of observed surface. With digital technology, remotely sensed data received a new organization level in form of matrix that contains the brightness values in its cells, what made this data suitable for statistical analysis and meaningful interpretation. In context of land use optimization, the data types developed in remote sensing can provide an effective basis for optimization. Essentially, the matrices and pixels can represent distribution of resources throughout the area, and even a pixel still generalizes this distribution within itself it is a matter of technology development to have pixels representing smaller and smaller portions of Earth surface. Therefore, pixel can be considered as “infinitely” small portion of area.

Numerous techniques have been developed to detect land characteristics and classify the values of brightness reflected by some area in different portions of spectrum. These techniques are divided, generally, into supervised and unsupervised classifications (Jensen, J. R., 1996), where in supervised one, the analyst develops spectral signatures of certain characteristic and than using a classification algorithm analyses every pixel assigning it the meaningful class value. It has become a tradition in remote sensing to analyze every pixel individually in an image in order to come up with class assignment. This way seems rational in terms of spectral analysis, as from its point of view a characteristic must be found within one pixel (Jensen, J. R., 1996). However, from the point of view of resource distribution and land use the above statement appears too narrow to see the composite recourses of multiple pixels which together (combining their
resources) can be developed as certain land use or LUT. The “pure pixel” problem becomes an obstacle for analysis when it considers just one unit of area that must have all the necessary characteristics in order to be classified as suitable one. Though, these limitations can be overcome.

The traditional representation of one “pure” pixel can be extended over multiple pixels that can in their composite value be assigned to certain class.

![Figure 1.7. Overcoming Pure Pixel Problem.](image)

In terms of values of resource availability in Figure 7.b the values of different pixels would not probably fall into one class according to the procedure of supervised classification. However, the composite value of those pixels will perfectly fit to the class developed from one pixel in Figure 7.a. Using such an approach one can find all possible combinations throughout the image where the composite value of pixels would fit for certain class, regardless of the shape of final area under this class. This principle is almost identical to the one used in the model were the overall performance per unit of area is the only factor that matters.
The problem of representation of resource distribution within one band of data has been solved already with researches on indices in remote sensing. There have been developed such indices as NDVI, MPDI, soil moisture index derived from microwave reflection (Wigneron, J.-P. at all, 2003). Generally, these indices describe some biophysical characteristics of surface they were detected from, and show the condition of these nature components. The nature of indices in remote sensing is similar to the nature of digital imagery, what means that the values of index are expressed in the same range and therefore may be considered in the classification process (Jensen, J. R., 1996). All together the properties of indexes in remote sensing make them perfectly suitable for economic evaluation of land.

As it is stated by Naveh and Liberman (1994 p.207) an economic land evaluation should investigate “environmental factors serving for the general public good”. Therefore, the land should be viewed as a system of factors or variables describing specific features, which may be used for land utilization. Considering the sustainability of given LUT on the land, the system should take into account the “cause-effect relationship between landscape pattern and variable of interest” (Turner M.G., 2001. p 290). The approach, which can meet these requirements, is used by landscape ecology to determine the spatial landscape structure and with different level of “harmony [fitness]” between biota and natural conditions (Haines-Young R., 1994 p. 43). Within change of focus from vegetation to LUT, this approach is becoming useful for EEL. With such a consideration, the parallel between landscape ecology methods and digital image classification in remote sensing may be drawn. Taking the LUC as parameters of land use development, it is becoming possible to determine the most suitable land use or LUT for particular area.
Basing on geo-statistical data of indices representing levels of resource availability, the signatures of land use can be developed as it is done in remote sensing. However, the land use signatures should be designed not in form of mean and standard deviation, but in form of sufficient minimum normalized by carrying capacity for a LUT (Lein, J. K., 1995, p. 23). Use of mean values for land use signatures loses its sense because of its inability to catch the perspective side of development. In standard classification process a pixel falls into class only if its value belongs to the domain limited by standard deviation from mean. But, it is just a matter of time or intensity for the LUT to run out of resources. Therefore, the sufficient resource minimum is becoming a basis for the classification that should determine whether the land map unit has enough resources to start the development of the LUT, and if it has more than minimum, the LUT gives the perspective of sustainable development and proper utilization of the area.

<table>
<thead>
<tr>
<th>Resource 1</th>
<th>Resource 2</th>
<th>Resource 3</th>
<th>Resource 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>sufficient minimum</td>
<td>100</td>
<td>80</td>
<td>50</td>
</tr>
<tr>
<td>real values</td>
<td>120</td>
<td>90</td>
<td>60</td>
</tr>
</tbody>
</table>

Figure 1.8. Signatures and Factual Data in the Model.
Figure 3 illustrates a possible situation when the theoretical area has more resources than sufficient minimum. It is clear, then, that the area can be used for this LUT (LUT suits for the area) having a surplus of resources 1, 2 and 3. This would bring higher performance level for this particular land use.

According to the stated above, the LMU may be described as a one-dimensional array of resources $R = (LC1, LC2, LC3, ... LCn)$ or as a set of LC values $R=\{LC\}$. The land utilization type may also be described as a set or array of land use requirements $LUT = (LUR1, LUR2, LUR3, ... LURN)$ or $LUT = \{LUR\}$. Thus, the process of evaluation can be narrowed down to comparison of responding land characteristics and land use requirements. Also, the complex evaluation can be completed as a comparison of combined LURs ($LUT1 + LUT2 + ... LUTn$) with LCs of $R$. Within this, the LMU may be tested for possibility to provide the development of several land utilization types in the same time.

To find the area on the image (matrix) that would suit for the certain LUT, actual values of LMUs/pixels in the image must be compared with signature and then assigned to the class. However, as it was pointed out, the LUT can be adopted on multiple pixels and it is not necessary for every pixel to satisfy the conditions of sufficient resource minimum. The factual data from the area can be reconfigured (the number of participating pixels can be changed) and than compared with signature. This approach is frequently used in Landscape Ecology while determining grain size of “the area in the landscape occupied by the object of interest” (Plotnik et al, 1993). The method is called Lacunarity Analysis and was brought into the field Landscape Ecology by Plotnik and colleagues in 1993.
The procedure lacunarity analysis involves sliding window/box technique, where the window is moved through the grid with step of one row and one column at a time. The size of the window increases after each full iteration up till when the size of moving window equals size of image. During the analysis, every pixel is sampled several times being involved in composite values of cell groups (windows). For landscape ecologists the results would be the plot of natural log of lacunarity value and size of the window that describes the particularity of the object’s spatial distribution (Bastin, G. N. at all, 2002).

Reflecting “the distribution of gap or cluster sizes in data” (Weishampel, J. F., at al, 2001) the procedure of lacunarity analysis helps to determine the scale of study and cluster the landscape statically what reduces time for further analysis. Also, it depicts some useful information about texture of the landscape that can show changes in patterns of land use distribution (Cain D. H., Riitters R., & Orvis K., 1997). Even though the process involves tremendous number of elementary operations, it presented significant advantages over static landscape texture analysis, since it uses just 3x3 (most common) pixel window as constant texture number for determining fractal dimensions in landscape (Chen, K., Jacobson, C., Blong, R., 2001). The concept of dynamic, (changing size and shape) window helps to catch the landscape texture even finer, what can contribute in work of environmental planners.

The problem with use of lacunarity analysis in land use planning and optimization would be the fact that environmental planners do not deal with predetermined “yes” or “no” data; the choice is not made yet, and the problem does not lay in just optimal clustering of the map. The advantage, though, would be that the levels of future
productions and resources needed are known. Hence, this can be used in creating new semi-lacunarity analysis procedure that would make this choice and cluster the map all at the same time. The approach supports the model described above, and offers significant flexibility for each individual case of optimization, where numerous options of land utilization types are to be considered.

Many authors in the field of environmental planning agree that environment determines the “opportunities to be exploited” (Lein J.K., 2003), so the land evaluation must offer the opportunity to catch all possible combinations of pixels (LMUs) that may be used for a LUT assessment and then choose the optimal one. This combined approach of Landscape Ecology and Remote Sensing to the EEL presents the advantage of easy algorithmisation of the process. This approach allows putting all the LCs into database and comparing those with LURs using means of simple matrix analysis. This may generate evaluation data with indexation to LMU what gives the information to decision makers about what can be improved in the area, in other words, planners can have the access to the "trade-offs" of the area.
Chapter 2. Methodology

Development of theoretical backgrounds of the model brings the outline for the model practical implementation and testing. In modern days such implementation cannot avoid the use of computers. Even from the look, the procedure involves enormous number of operation that can hardly be done even on relatively small problem sizes. The first attempt to computerize FAO methodology of land use evaluation was done by the team of scientists in Cornell University lead by Dr. David G. Rossiter. In their Automated Land Evaluation System (ALES) they used liner optimization (simplex method) to evaluate performance and level of sustainability for the land use; where the map units of the area were analyzed and assigned their utilization type one by one. The software proved to be effective, but development of GIS recapture the initiative from ALES and the project was frozen in development in 1996. Having such powerful competitors, the alternative model presented in this study must be implemented as computer program. Though, it must remain simple and open for modification for other possible uses of the methodology.

2.1. Implementing the Model

To visualize the problem better we can consider a hypothetic task of a forest manager that needs to re-grow the forest on a clear-cut. He has information about several soil characteristics distributions on the clear-cut parcel (in form of maps/matrices in our case) and knows the requirements to these characteristics that would allow planting the best quality trees (for different trees – different set of requirements). Relating the situation to the model, the forest manager must keep in mind that, the utilization of land will
always give different levels of performance on different land map units, but there are always some levels of feasibility that make it possible for the land utilization type to be developed on the area. It would be a simple but not enough rational decision to develop the land use just on each map unit that has more than sufficient level of resources and leave all the rest without development. The level of output (tree growth) on those map units will be always more than effective. Therefore, it can compensate for development of other units that have lower level of performance.

The model suggests that, if the forest manager receives in average 10 cubic meters of timber form 100 map units and the sufficient level of productivity is 8 cubic meters, he or she can put the trees on some other less effective areas such that it would give her less in average but more in over all volume. For example as in situation described above, he can receive (10*100) 1000 cubic meters of timber but if he plants these trees on 150 map units with average of 8 cubic meters per unit it will give the output of 1200 m$^3$. Hence, the tree production will receive more returns and will utilize the land more efficiently.

Additionally, the model can incorporate the factor of location that has been always the problem for tree planting. Since the resources are spread “randomly” it can occur that trees will be dispersed all over the parcel without direct access to each other. This factor increases the cost of tree planting (transportation expenses) and decreases its economic effectiveness. To ensure future access the manager must keep the trees of one type within one patch what would eliminate unnecessary transportation expenses and minimize negative impact on surrounding trees that are not ready jet to be cut for future utilization. Incorporating these factors in the process of forest planning will improve the
sustainability of this land use in the area though keeping commercial feasibility of the tree planting and taking into account natural conditions (resources) of the land.

To accomplish this task the manager must find the core area where the trees can grow the best and try to add surrounding areas checking whether there are still enough resources to support the growth of trees with proper overall performance level. The search for core area would involve the semi-lacunarity analysis that has the same procedure as lacunarity analysis but instead of looking for the proper size of cells for finding patches (scale problem) the semi-lacunarity would look for the areas that would meet the requirements of considered signatures no matter what size or form it would be. Literally, it would be simply going over the clear-cut parcel making subdivisions of $n$ by $m$ rectangles and comparing the mean characteristics values of subdivided patch with requirements that have to be met, and if it is the case collecting all options and then changing the size and form of the subdivisions for further processing. Though, the procedure of lacunarity analysis has the limitation of patch/subdivision shape – rectangular/square. Therefore, naturally, to find some other options of shape, the search should be continued in the edge of the core area – within pixels/cells of initial size that are adjacent to the core area. The manager than should pick and include into core area the best cells that would preserve or improve the situation with meeting the requirements and the process should allow him to reconsider the form and edge of the patch after every accomplished choice.

The goal of the algorithm is to spread the land use on the maximum possible area or in other words to involve maximum number of pixels for it, such that the mean amount of resource would be more than or equal to the given land use signature. Together with
this, the condition of compactness should be satisfied, in other words the resources’ use should be localized within one body of land use type to be self sustained.

In abstract terms we can denote our input data as matrices e.g. -

A(i, j), B(i, j), C(i, j), D(i, j); -- i and j are the coordinates of pixels

The \textit{land use signature} we can denote than as a vector: \( s_0\{a_0, b_0, c_0, d_0\} \);

A pixel or land Use unit we can denote as \( x \) with coordinates I and j \( (x_{ij}) \) that have values correspondingly in each matrix (band) \( x_{ij}\{a_{ij}, b_{ij}, c_{ij}, d_{ij}\} \)

The Body of future land use parcel we can denote as a set of \( n \) selected pixels that than can be denoted as where all of the selected pixel are adjacent to each other and the mean values for each matrix for whole set are more than or equal to the corresponding value of the signature

\( x_m\{a_m, b_m, c_m, d_m\} \)

where \( a_m=(\sum_{n=1}^{n} x_n\{a\})/n \); \( b_m=(\sum_{n=1}^{n} x_n\{b\})/n \) …

The objective function of the algorithm would be to maximize \( n \) of the set, what would be a subject to the following constraints

\[
\begin{align*}
 a_m &\geq a_0; & c_m &\geq c_0; \\
 b_m &\geq b_0; & d_m &\geq a_0;
\end{align*}
\]

more than or equal to signature values

The adjacency of pixels (as a constraint) can be denoted as follows:

Suppose we have the pixel which is already in the body of the land use \( (x_{i_0,j_0}) \) is a subset of \( X_n \) an than we can say that in order to be adjacent the \( i_c \) and \( j_c \) coordinates of a candidate \( (x_c\{i_c, j_c\}) \) should be

\( |i_0 - i_c| = 1; \quad |j_0 - j_c| = 1; \)
and consequently:

\[ x_m \{a_m, b_m, c_m, d_m\} + x_c \{a_c, b_c, c_c, d_c\} \geq s_0 \{a_0, b_0, c_0, d_0\}; \]

**meaning:**

\[ a_m + a_c \geq a_0; \]
\[ b_m + b_c \geq b_0; \]
\[ c_m + c_c \geq c_0; \]
\[ d_m + d_c \geq a_0; \]

As it is seen from the description above, the problem is not linear and bares more of a heuristic character.

### 2.1.1. The Structure of Objects in the Procedure

In order to be efficient and well organized every computerized procedure must have a structured system of participating objects and data types (Jähne, B., 1997). The research included design of functional system of Abstract Data Types (ADT) that would participate in semi-lacunarity analysis as well as in functional neighboring procedures (edge search). The ADT system will consist of:

- **Basic Matrix ADT** - bands initialized from source file (MTX type in the code in Appendix A);
- **Pixel** – area representing unit, holding values of resource bands for the area (Pix type in appendix a);
- **Matrix of Pixels** – initialized from Basic Matrix, gives more efficiency and flexibility in manipulations than Basic Matrix, holds Pixels ADT inside rather than just values what simplifies understanding and future interpretation of the results (MPX type Appendix A)
• Signature ADT – has a form of one dimensional array of integers as denoted above, with overloaded procedures of comparison and calculation of improvement factor;

• Window ADT - contains four manipulation characteristics: x- horizontal dimension of the window, y – vertical dimension of the window, and two starting point coordinates –start_x and start_y;

• Patch ADT - (structured core area) initialized from previously done semi-lacunarity analysis (multiple bodies must be possible), having internal data members Body and Edge.

2.1.2. The Algorithm: Procedures

The algorithm is organized in 2 steps:

Step 1 Semi-lacunarity Analysis

Initialization of Basic Matrix from source file and conversion to the Matrix of Pixels,

Initialization of Signatures and Window object to the size of generalization level.

Start of slicing Loop

While going over the Matrix the Window will collect mean values of band and those will be compared with signature and since all of the pixels within the window are adjacent the test for adjacency would be redundant. All the pixels of suiting window receive denotation that they are selected for this land use in a corresponding class member (Appendix A, Pixel.h). To avoid redundancy Windows that contain already selected pixels are not analyzed since this can also destroy the balance of future Patches (ability to comply with “>=” requirements).
Step 2 – Functional Neighboring (Edge Search)

The Patches are initialized with their internal structure (Body and Edge). The process will divide chosen pixels into groups of pixels, such that intersection of any of the Bodies would be the empty set and any of the Body pixels of two different Patches are not adjacent to each other. Meaning:

\[ x_{b1}(i_{b1}, j_{b1}) \] – pixel of Body 1 for all pixels of Bodies 1 and 2
\[ x_{b2}(i_{b2}, j_{b2}) \] – pixel of Body 2
\[ |i_{b1} - i_{b2}| > 1; \]
\[ |j_{b1} - j_{b2}| > 1; \]

Also, the algorithm constructs the class member Edge for every Land Use Body that would represent all pixels adjacent to the Body in question. Than the algorithm will test every pixel in the Edge whether the pixel can be chosen for being added to the Body. Meaning:

Denoting Edge as set of \( e \) pixels adjacent to the Body -- \( X_e \{x_1, x_2, x_3, \ldots x_e\} \),

Pixel \( x_c \) – pixel candidate for being added to the body ( \( x_c \in X_e \));

Body -- \( X_m \) (same as described above in the text);

\[ x_m(\{a_m, b_m, c_m, d_m\}) + x_c(\{a_c, b_c, c_c, d_c\}) \geq s_0(\{a_0, b_0, c_0, d_0\}); \]

The pixel-candidate that complies with the test above and improves the availability of resources the most, becomes the member of the Body. After this the edge will be reinitialized according to changes in the Body and Step2 will be repeated until there will be no candidates that comply with the test above.
2.2. Analysis of Algorithm Performance

This part of the chapter will be focused on analysis of two main procedures involved for accomplishing the task. Those procedures are Semi-Lacunarity Analysis and Search in Edge or Functional Neighboring (last two names refer to the same procedure and will be used further interchangeably). The main measurement in here would be the number of pixels in one band, that in this particular case constitute 1000 by 1000 or simply a 1000000 pixel. So the complexity of the problem will be denoted using $n$ – number of pixels in matrix. Also, the analysis will be presented just for one involvement of a procedure.

2.2.1. Time complexity

a) Semi-lacunarity analysis

The procedure constitutes calculation of mean values of bands for pixels found inside the window that slices within the matrix of input data, changing its dimensions after every loop. It involves two types of manipulation: window size/dimension manipulation and starting address manipulation that slices the window through the matrix. Taking this into account the number of manipulations per one involvement of the procedure can be denoted with formula:

$$\sum_{hw=h}^{1} \sum_{ww=w}^{1} (h-hw+1)*(w-ww+1)$$
Where: h – height of the matrix
w – width of the matrix
hw – height of the window
hw – width of the window

$n$- therefore here will appear as $h \times w$
or in case with square –
h = w = $\sqrt{n}$

It is easy here to estimate the number of manipulations with window size 1 and 1000 but also the calculation of mean values would be evolved here and for this purpose we should multiply the formula by $hw$ and $ww$ each time in summation. This would give the number of elementary operations for both sizes of the window equal to 1000000. Taking this simple approach, the order would simply be $n^2$ (n*number of possible shapes of window |1000000 in this case|). However, the number of operations will vary throughout the procedure. To give an idea of what kind of numbers of elementary operations would appear, let’s calculate the number of slicing and mean calculating operations for window size 500 by 500: $501^2 \times 250000$ what is more than 62 billion operation and that would be maximum operations for this particular case. Analysis of dynamics of number of operation per one window loop shows that the highest number of operations would be on the half of matrix sizes window. Therefore it constitutes upper bound for this procedure. This size of window can also be denoted as $n/4$. Consequently, the estimated Order of the algorithm would be: $\frac{1}{4} n \times n^2$ that is Order of $n^3$. Additionally, the number of bands should be involved in calculation, since mean values must be calculated for each band. In this particular case the number of bands is 4 and it is not too big. Probably, the number of bands would not grow high even in really huge research projects. As the author can imagine, the number of bands may grow up to few dozens, but no more. Therefore, in this report the number of bands would be taken as a constant.
To accomplish the task of full semi-lacunarity analysis just for one signature for problem size 100000 it would take at most 1000000000000000000 (10^{18}) operations. With ability of modern super computers to produce 10^9 elementary operations per second this would take 10 billion seconds to accomplish, what is 2 *10^6 hours (317 years) and this is just for one signature. However, it is reasonable to say that rarely the signatures for analyses would occupy that big an area within the extent. So the algorithm gives a possibility to specify the size of starting window. This gives ability for user to be flexible with generalization level, which is the key factor in semi-lacunarity analysis. This research was conducted with generalization level 4: meaning the semi-lacunarity analysis procedure starts at window size 4 by 4 or 16 pixel square window. Also many programming optimizations were used to decrease the number of operations including avoidance of procedure repetitions on already tested and assigned pixels. 2.8 GHz Intel Pentium processor had accomplished the task of semi-lacunarity analysis in just ~1 hour.

b) Edge search

The procedure of edge search does not include complicated slicing and reshaping manipulations but still it has polynomial time of more than n. To denote the upper bound for this procedure, lets consider the worst case when the preceding semi-lacunarity analysis found just one pixel to be core area and eventually all the other pixels in matrix were one by one joined one by one to that body with the most incompact manner (the situation is impossible in real life but would illustrate the upper bound). With this context the search for initial pixel will take n operations plus initializing edge will take constant*n. With growth of body taking pixels one by one the number of operation will
grow similarly to dynamic of sum of first $k$ numbers in sequence: $\frac{1}{2} n (n-1)$ and this would be the upper bound. The dynamics of number of operations here behaves similarly to the semi-lacunarity analysis. The number of operations per cycle would be maximal when the body and edge reach half of the matrix size. Therefore the previously given formula should be modified to this form: $2^{*\frac{1}{2}} \left(\frac{1}{2} n \left(\frac{1}{2} n - 1\right) >> \frac{1}{2} n \left(\frac{1}{2} n - 1\right)\right)$ which is still order $n^2$.

Overall Time Complexity of the algorithm will constitute $n^3 + n^2$ what is a polynomial time that will round up to the order of $n^3$.

2.2.2. Space Complexity

This characteristic of the algorithm is not a serious problem since during all the procedure the computer will have in memory at most two copies of input matrix. Therefore the Space Complexity for the whole algorithm will constitute Order $n$ (Order of the input matrix size). However, there is still room for improvement. The 32 bit integers that were used to denote values of different characteristics in matrix may be easily replaced with 8 bit short integers that would decrease the memory usage by 4 times. While running in the current condition and given size of the input data algorithm was consuming at most 260 MB of memory and decreasing this number to 65 MB would give a significant improvement.

Giving the over all performance evaluation it is necessary to say that even with current state of computer technology this algorithm is feasible for being used in solving many problems not limited by only field of land use planning or forestry.
2.3. Strategies and Options in The Analysis

The algorithm offers many options for performing analysis that would highly depend on the ideas that users have in mind. The first and the most important factor/options in the analysis is the order of land uses (order of signatures in input file). By putting a particular land use before some other land uses the analyst gives a priority to the first land use over the others. This is happening because the memory of computer signatures are put in stack with FIFO (First In First Out) processing order, what means that the first land use will receive the whole parcel for analysis and all the other ones will develop on the remaining area. Therefore, by changing the order of signatures the picture of choice and placement of land uses can be changed.

On one hand this may become the limitation of the algorithm as it can bring different outputs for the same sets of land uses. When considering natural vegetation distribution the problem of land use (vegetation community) order can be overcome by performing independent lacunarity analysis for each community and simply by counting assessed pixels the analyst can find out which vegetation type would dominate in the area and according to this she can sort the signatures in input file for the overall analysis with edge search.

However, the land-use order factor can be used to the advantage of the analyst. Depending on market conjecture, a farmer, for instance, can manipulate the order of crops to be harvested and give the maximum priority for the most valuable product. This will extend the area assessed to the crop and help the farmer to succeed in his or her business. In this study, the author used exactly this approach by taking prices from Ohio Timber Price Report of July 14, 2003 issued by Ohio Department of Natural Resources.
To demonstrate flexibility of the model in real situation, two strategies were chosen to analyze input data. The first one would model the situation where all land uses arrive at the study area at the same time and have equal opportunity to develop their habitats (patches). Connecting this to the object of study, the forest types would be given their core area immediately and than they will develop in their edges without altering each other’s patch shapes. This strategy resembles more of a natural way of forest development.

Second strategy models the situation where forest planner puts priority on growth of some trees. Consequently, the forest type with highest priority receives the possibility to have core area and edge development prior to the others. This means that LUT of lower priority receives only the area remaining after the development of land uses of higher priority. To show differences in procedures lets take two land use types and simulate the strategies 1 and 2.
Figure 2.1. Differences in Strategies 1 and 2.
As it is seen from Figure 2.1 the order of semi-lacunarity analysis and edge search in Strategy 1 is different from that of Strategy 2. Even though both of strategies detected same areas of land uses (for both together) what means that equal amount resources will be utilized by two land uses, Strategy 2 redistributed resources in favor of the first land use. In practice it can help a manager to harvest more valuable products on larger areas. This will keep the land use close to natural conditions and at the same time increase economic survivability/sustainability of the area.

The options may vary in different parts of the procedure. As it was mentioned above, the analyst may choose his or her level of generalization depending on the size of the input data and time given for the research. In this study the level of generalization (size of the starting window) was put to 4 (16 pixels). Also, if the analyst believes that the edge width is more than one and even with some gaps in adjacency the patch would still constitute one structural unit, the option of changing width of the edge should not be taken of the analysis. Though, in the written algorithm the size of the edge is hard coded, and to change the option the code should be altered and recompiled.

Finally, the test conditions of mean values comparing with signature can be modified to meet the requirements of minimums and maximums of band values in pixels to be joined to bodies. In case of this study, the test was not just comparison of values. The program calculates also the level of improvement to distinguish between pixels and choose the best candidate. The code is open for these modifications and therefore its ability to give proper results in analysis is not limited by this study.
Chapter 3: Tests and Results

For the purpose of algorithm testing, the study area was selected to model the hypothetic land use evaluation. In this chapter the forest management scenario will be developed and visualized in maps. Still, the task of the forest manager is to optimize forest growth in a manner that would exclude ineffective utilization of land resources, but prior to this the procedure must find these areas in order to see the difference after optimization.

3.1. Study area

The study area is located in the North corner of Athens County that is also a South Eastern part of Wayne National Forest – Athens division, located in Southeastern part of Ohio. Dimensions of the area can be denoted as a square of 18178 by 18178 feet (5542 by 5542 meters) with square of ~3 km². The area is a resent acquisition of Wayne national forest, and for almost 80% of the area the administration has not yet conducted any survey to come up with a management plan. This study is dedicated to help with design and development of such a plan.

Figure 3.1. Location of the Study Area.
For the purpose of research that would concentrate on the resources for the forest grows, it is essential to have data describing the soil properties and terrain characteristics of the area. The reason to involve soil physical properties in the research is that those constitute main basis for a tree growth influencing correspondingly:

**Depth of Soils** – development of root system

**Water availability in Soil** – hydrological conditions of a tree growth


Also the terrain data (slops) play significant role in development of root system and determination of hydrological regime for the area and consequently for the forest growth.

**Table 1. Physical Properties of Soils Found within the Study Area (Source: USDA, Natural Resources Conservation Service. Template Database, Version 30).**

<table>
<thead>
<tr>
<th>Soil Map Code</th>
<th>Depth (in)</th>
<th>Availability of water (in³/in³)</th>
<th>Organic Matter (in³/in³)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AdC2</td>
<td>30</td>
<td>0.12</td>
<td>0.5</td>
<td>0.5 Aaron-Gilpin complex, 6 to 12 percent slopes, eroded</td>
</tr>
<tr>
<td>BkD</td>
<td>20</td>
<td>0.12</td>
<td>0.3</td>
<td>Berks-Westmoreland silt loams, 15 to 25 percent slopes</td>
</tr>
<tr>
<td>BkE</td>
<td>22</td>
<td>0.13</td>
<td>0.2</td>
<td>Berks-Westmoreland silt loams, 25 to 40 percent slopes</td>
</tr>
<tr>
<td>BkF</td>
<td>20</td>
<td>0.08</td>
<td>0.3</td>
<td>Berks-Westmoreland silt loams, 40 to 70 percent slopes</td>
</tr>
<tr>
<td>BoD</td>
<td>43</td>
<td>0.08</td>
<td>0.1</td>
<td>Bethesda shaly silty clay loam, 8 to 25 percent slopes</td>
</tr>
<tr>
<td>BoE</td>
<td>35</td>
<td>0.09</td>
<td>0.1</td>
<td>Bethesda shaly silty clay loam, 25 to 40 percent slopes</td>
</tr>
<tr>
<td>BoF</td>
<td>36</td>
<td>0.09</td>
<td>0.1</td>
<td>Bethesda shaly silty clay loam, 40 to 70 percent slopes</td>
</tr>
<tr>
<td>BrC</td>
<td>32</td>
<td>0.08</td>
<td>0.1</td>
<td>Brookside silt loam, 8 to 15 percent slopes</td>
</tr>
<tr>
<td>BrD</td>
<td>21</td>
<td>0.11</td>
<td>0.8</td>
<td>Brookside silt loam, 15 to 25 percent slopes</td>
</tr>
<tr>
<td>BrE</td>
<td>31</td>
<td>0.11</td>
<td>0.4</td>
<td>Brookside silt loam, 25 to 40 percent slopes</td>
</tr>
<tr>
<td>BtB</td>
<td>28</td>
<td>0.08</td>
<td>0.3</td>
<td>Bethesda channery loam, 0 to 8 percent slopes</td>
</tr>
<tr>
<td>BtC</td>
<td>28</td>
<td>0.08</td>
<td>0.3</td>
<td>Bethesda channery loam, 8 to 20 percent slopes</td>
</tr>
<tr>
<td>BtE</td>
<td>32</td>
<td>0.07</td>
<td>0.2</td>
<td>Bethesda channery loam, 20 to 40 percent slopes</td>
</tr>
<tr>
<td>BtF</td>
<td>29</td>
<td>0.08</td>
<td>0.3</td>
<td>Bethesda channery loam, 40 to 70 percent slopes</td>
</tr>
<tr>
<td>Cd</td>
<td>35</td>
<td>0.14</td>
<td>0.5</td>
<td>Chagrin loam, rarely flooded</td>
</tr>
<tr>
<td>Cg</td>
<td>36</td>
<td>0.14</td>
<td>0.5</td>
<td>Chagrin silt loam, frequently flooded</td>
</tr>
<tr>
<td>CmC</td>
<td>34</td>
<td>0.08</td>
<td>0.5</td>
<td>Clymer loam, 8 to 15 percent slopes</td>
</tr>
<tr>
<td>DkE</td>
<td>16</td>
<td>0.1</td>
<td>0.5</td>
<td>Dekalb loam, 25 to 40 percent slopes</td>
</tr>
<tr>
<td>Soil Code</td>
<td>ID</td>
<td>Slope</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>----</td>
<td>-------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>DtF</td>
<td>29</td>
<td>0.16</td>
<td>0.5 Dekalb-Westmoreland complex, 40 to 70 percent slopes</td>
<td></td>
</tr>
<tr>
<td>EcA</td>
<td>37</td>
<td>0.19</td>
<td>3 Euclid silt loam, rarely flooded</td>
<td></td>
</tr>
<tr>
<td>FaD</td>
<td>22</td>
<td>0.07</td>
<td>0.3 Fairpoint silt loam, 8 to 25 percent slopes</td>
<td></td>
</tr>
<tr>
<td>FbE</td>
<td>18</td>
<td>0.07</td>
<td>0.3 Fairpoint shaly clay loam, 25 to 40 percent slopes</td>
<td></td>
</tr>
<tr>
<td>FbF</td>
<td>32</td>
<td>0.07</td>
<td>0.3 Fairpoint shaly clay loam, 40 to 70 percent slopes</td>
<td></td>
</tr>
<tr>
<td>FaC</td>
<td>43</td>
<td>0.17</td>
<td>0.7 Fitchville silt loam, 0 to 3 percent slopes</td>
<td></td>
</tr>
<tr>
<td>GfB</td>
<td>36</td>
<td>0.16</td>
<td>2 Glenford silt loam, 2 to 6 percent slopes</td>
<td></td>
</tr>
<tr>
<td>GmA</td>
<td>42</td>
<td>0.16</td>
<td>0.8 Glenford silt loam, 0 to 3 percent slopes</td>
<td></td>
</tr>
<tr>
<td>GmB</td>
<td>32</td>
<td>0.17</td>
<td>0.8 Glenford silt loam, 3 to 8 percent slopes</td>
<td></td>
</tr>
<tr>
<td>GmC</td>
<td>27</td>
<td>0.06</td>
<td>0.9 Glenford silt loam, 8 to 15 percent slopes</td>
<td></td>
</tr>
<tr>
<td>GsB</td>
<td>33</td>
<td>0.12</td>
<td>0.3 Guernsey silt loam, 3 to 8 percent slopes</td>
<td></td>
</tr>
<tr>
<td>GsC</td>
<td>33</td>
<td>0.12</td>
<td>0.3 Guernsey silt loam, 8 to 15 percent slopes</td>
<td></td>
</tr>
<tr>
<td>GuC</td>
<td>38</td>
<td>0.12</td>
<td>0.2 Guernsey-Upshur complex, 8 to 15 percent slopes</td>
<td></td>
</tr>
<tr>
<td>GuD</td>
<td>38</td>
<td>0.13</td>
<td>0.2 Guernsey-Upshur complex, 15 to 25 percent slopes</td>
<td></td>
</tr>
<tr>
<td>GwC</td>
<td>41</td>
<td>0.08</td>
<td>0.3 Guernsey-Westmoreland silt loams, 8 to 15 percent slopes</td>
<td></td>
</tr>
<tr>
<td>GwD</td>
<td>31</td>
<td>0.12</td>
<td>0.4 Guernsey-Westmoreland silt loams, 15 to 25 percent slopes</td>
<td></td>
</tr>
<tr>
<td>GwE</td>
<td>29</td>
<td>0.12</td>
<td>0.4 Guernsey-Westmoreland silt loams, 25 to 40 percent slopes</td>
<td></td>
</tr>
<tr>
<td>HcA</td>
<td>30</td>
<td>0.16</td>
<td>3 Hackers silt loam, 0 to 3 percent slopes</td>
<td></td>
</tr>
<tr>
<td>LkC</td>
<td>34</td>
<td>0.12</td>
<td>0.2 Licking silt loam, 8 to 15 percent slopes</td>
<td></td>
</tr>
<tr>
<td>LrF</td>
<td>26</td>
<td>0.12</td>
<td>0.3 Lowell-Gilpin complex, 35 to 70 percent slopes</td>
<td></td>
</tr>
<tr>
<td>Mh</td>
<td>40</td>
<td>0.21</td>
<td>0.6 Melvin silt loam, frequently flooded</td>
<td></td>
</tr>
<tr>
<td>Nn</td>
<td>33</td>
<td>0.2</td>
<td>1.4 Newark silt loam, frequently flooded</td>
<td></td>
</tr>
<tr>
<td>No</td>
<td>35</td>
<td>0.2</td>
<td>1.4 Nolin silt loam, frequently flooded</td>
<td></td>
</tr>
<tr>
<td>Or</td>
<td>28</td>
<td>0.17</td>
<td>0.7 Orrville silt loam, frequently flooded</td>
<td></td>
</tr>
<tr>
<td>OtB</td>
<td>33</td>
<td>0.07</td>
<td>0.3 Omulga silt loam, 3 to 8 percent slopes</td>
<td></td>
</tr>
<tr>
<td>OtC</td>
<td>33</td>
<td>0.07</td>
<td>0.3 Omulga silt loam, 8 to 15 percent slopes</td>
<td></td>
</tr>
<tr>
<td>RcC</td>
<td>25</td>
<td>0.13</td>
<td>0.7 Richland loam, 8 to 15 percent slopes</td>
<td></td>
</tr>
<tr>
<td>RcD</td>
<td>25</td>
<td>0.13</td>
<td>0.9 Richland loam, 15 to 25 percent slopes</td>
<td></td>
</tr>
<tr>
<td>RcE</td>
<td>25</td>
<td>0.13</td>
<td>0.9 Richland loam, 25 to 40 percent slopes</td>
<td></td>
</tr>
<tr>
<td>UpC</td>
<td>30</td>
<td>0.12</td>
<td>0.2 Upshur silt loam, 8 to 15 percent slopes</td>
<td></td>
</tr>
<tr>
<td>UpD</td>
<td>30</td>
<td>0.12</td>
<td>0.2 Upshur silt loam, 15 to 25 percent slopes</td>
<td></td>
</tr>
<tr>
<td>UsD</td>
<td>31</td>
<td>0.12</td>
<td>0.2 Upshur-Elba silt loam, 15 to 25 percent slopes</td>
<td></td>
</tr>
<tr>
<td>VbD</td>
<td>32</td>
<td>0.13</td>
<td>0.4 Upshur-Steinsburg complex, 15 to 25 percent slopes</td>
<td></td>
</tr>
<tr>
<td>WdB</td>
<td>26</td>
<td>0.19</td>
<td>0.8 Wellston silt loam, 3 to 8 percent slopes</td>
<td></td>
</tr>
<tr>
<td>WdC</td>
<td>26</td>
<td>0.19</td>
<td>0.9 Wellston silt loam, 8 to 15 percent slopes</td>
<td></td>
</tr>
<tr>
<td>WeB</td>
<td>28</td>
<td>0.17</td>
<td>0.8 Westmore silt loam, 3 to 8 percent slopes</td>
<td></td>
</tr>
<tr>
<td>WeC</td>
<td>28</td>
<td>0.17</td>
<td>0.8 Westmore silt loam, 8 to 15 percent slopes</td>
<td></td>
</tr>
<tr>
<td>WhC</td>
<td>26</td>
<td>0.14</td>
<td>0.3 Westmoreland-Guernsey silt loams, 8 to 15 percent slopes</td>
<td></td>
</tr>
<tr>
<td>WhD</td>
<td>29</td>
<td>0.15</td>
<td>0.3 Westmoreland-Guernsey silt loams, 15 to 25 percent slopes</td>
<td></td>
</tr>
<tr>
<td>WhE</td>
<td>28</td>
<td>0.16</td>
<td>0.2 Westmoreland-Guernsey silt loams, 25 to 40 percent slopes</td>
<td></td>
</tr>
<tr>
<td>WhF</td>
<td>30</td>
<td>0.15</td>
<td>0.2 Westmoreland-Guernsey silt loams, 40 to 70 percent slopes</td>
<td></td>
</tr>
<tr>
<td>WkF</td>
<td>32</td>
<td>0.16</td>
<td>0.5 Westmoreland-Guernsey silt loams, benched, 40 to 70 percent slopes</td>
<td></td>
</tr>
<tr>
<td>WmE</td>
<td>32</td>
<td>0.17</td>
<td>0.5 Westmoreland-Upshur complex, 25 to 40 percent slopes</td>
<td></td>
</tr>
<tr>
<td>WtC</td>
<td>26</td>
<td>0.19</td>
<td>0.9 Woodsfield silt loam, 8 to 15 percent slopes</td>
<td></td>
</tr>
<tr>
<td>ZnB</td>
<td>33</td>
<td>0.1</td>
<td>0.3 Zanesville silt loam, 3 to 8 percent slopes</td>
<td></td>
</tr>
</tbody>
</table>

- **Note.** Soil data corresponds to values of the characteristics on the middle of the soil profile, as the main development of root systems would occur there.
The soils were distributed throughout the study area as follows from the Figure below:

**Figure 3.2. Distribution of Soil Types on the Study Area** (Source: http://www.athensgis.com).
Terrain characteristics were derived on the basis of 5 foot contour line elevation map (source: http://www.athensgis.com) of the area using ESRI ARCMAP-Spatial Analyst.

Figure 3.3. Terrain Characteristics of the Area: (a) Elevation; (b) Slope.
3.1.1. Input Data and Preparations

To minimize the biases from generalizations made in soil maps the input data must be reinterpreted in raster format but not directly. The data should be smoothed to resemble the natural resource distribution. Therefore, the interpolation procedure was chosen as a mean of overcoming generalizations. For this purpose, the grid of points was designed with taking into account scale and quality of materials presented above. In this context the grain of the input matrices was estimated as square of 5.5 by 5.5 meters (30 m$^2$) that would produce matrix of 1000 by 1000 pixels, with each pixel representing 2-3 trees. To minimize the effect of interpolation and to catch maximum details on soil map the grid of points was designed with dimensions of 50 by 50 in between point spaces in order to have just 20 pixels between two points for further interpolation.

Figure 3.4. Grid of Points For Interpolation.
Data values to the points were brought via Selection by Location – which points are located within borders of certain soil type. Doing this, the soil type data were assigned to the points and then the joined table brought to the points the values of Physical Soil Properties presented in Table 1. Also, the soil data as well as slope data were re-stretched according to formula:

\[ 256 \times \frac{PV}{MV} \]

Where: MV – Maximum value of the given characteristic
PV – Actual value of the characteristic in given point

This re-stretching allowed to have all the characteristics in a common scale and from this moment the interpolated data would look more like a monochromatic image – and if many they would represent data bands rather than grids of different scales and types of values. Additionally, for the purpose of input simplification and elimination of none value pixels, the grid of points and slope data were turned ~8 degrees westward. After these preparations the data values were interpolated resulting following situation.

Figure 3.5.1 Input data: a) – Slope; b) – Depth of Soil.
As it is seen form figures 3.6.1 and 3.6.2 the interpolation affected the pictures of Soil data distribution, but despite this condition it gave the data matrix smooth enough for the analysis (clearly that interpolation effect will influence the final result of the study). All data bands were converted into ASCII format and stored into the input file, with format of

- first non comment line denotes correspondingly number of bands and dimensions of matricides;

- all the remaining non comment lines give actual values information

- comment lines start with symbol “%”

### 3.1.2 Deriving Signatures

Since the forest was chosen as the object of study, naturally the forest types would be the LUTs of the interest for this research. Though the database, received form Wayne National Forest Administration, has more than 30 types of forest stands, all of them can be grouped into 4 main groups selected to be the LUTs.
1. **Maple Dominated Forest**

2. **Mixed Hardwoods**

3. **Oak Dominated Forest**

4. **Pine Dominated Forest**

Signatures were derived via making Selection by Attributes in Forest Stand Data. The first criteria for selection where all stands that are more than 70 years old and have Condition Class less than 3 (Condition Class denotes quality of forest stand ranking it from 1 as highest value and 9 as lowest value). From this selection, the stands were grouped by dominating tree type and using Selection by Location, the minimum requirements of their growth conditions were estimated. In order to show both efficient and inefficient uses of the area, the signatures were developed for condition class 1-3 – feasible for the model, and class 4-6 – infeasible one.

**Table 2.1. Signatures for Forest Types of Condition Class 1-3 and Age more than 70.**

<table>
<thead>
<tr>
<th>Properties\Forest Type</th>
<th>Maple Dominated Forest</th>
<th>Mixed Hardwoods</th>
<th>Oak Dominated Forest</th>
<th>Pine Dominated Forest</th>
</tr>
</thead>
<tbody>
<tr>
<td>Depth of Soil Profile</td>
<td>29</td>
<td>32</td>
<td>24</td>
<td>16</td>
</tr>
<tr>
<td>Water available</td>
<td>0.17</td>
<td>0.14</td>
<td>0.1</td>
<td>0.08</td>
</tr>
<tr>
<td>Organic matter</td>
<td>0.8</td>
<td>0.6</td>
<td>0.7</td>
<td>0.3</td>
</tr>
<tr>
<td>Slope in %</td>
<td>0.28</td>
<td>0.13</td>
<td>0.29</td>
<td>0.11</td>
</tr>
</tbody>
</table>

**Restretched Values**

<table>
<thead>
<tr>
<th>Properties\Forest Type</th>
<th>Maple Dominated Forest</th>
<th>Mixed Hardwoods</th>
<th>Oak Dominated Forest</th>
<th>Pine Dominated Forest</th>
</tr>
</thead>
<tbody>
<tr>
<td>Depth of Soil Profile</td>
<td>172</td>
<td>190</td>
<td>142</td>
<td>94</td>
</tr>
<tr>
<td>Water available</td>
<td>206</td>
<td>170</td>
<td>121</td>
<td>97</td>
</tr>
<tr>
<td>Organic matter</td>
<td>67</td>
<td>50</td>
<td>59</td>
<td>25</td>
</tr>
<tr>
<td>Slope</td>
<td>72</td>
<td>33</td>
<td>74</td>
<td>27</td>
</tr>
</tbody>
</table>
Table 2.2. Signatures for Forest Types of Condition Class 6-4 and no Age Limitation.

<table>
<thead>
<tr>
<th>Properties\Forest Type</th>
<th>Maple Dominated Forest</th>
<th>Mixed hardwoods</th>
<th>Oak Dominated Forest</th>
<th>Pine Dominated Forest</th>
</tr>
</thead>
<tbody>
<tr>
<td>Depth of Soil Profile</td>
<td>25</td>
<td>28</td>
<td>21</td>
<td>14</td>
</tr>
<tr>
<td>Water available</td>
<td>0.14</td>
<td>0.12</td>
<td>0.08</td>
<td>0.07</td>
</tr>
<tr>
<td>Organic matter</td>
<td>0.72</td>
<td>0.56</td>
<td>0.62</td>
<td>0.24</td>
</tr>
<tr>
<td>Slope in %</td>
<td>0.25</td>
<td>0.12</td>
<td>0.26</td>
<td>0.05</td>
</tr>
</tbody>
</table>

Restretched Values

<table>
<thead>
<tr>
<th>Properties\Forest Type</th>
<th>Maple Dominated Forest</th>
<th>Mixed hardwoods</th>
<th>Oak Dominated Forest</th>
<th>Pine Dominated Forest</th>
</tr>
</thead>
<tbody>
<tr>
<td>Depth of Soil Profile</td>
<td>148</td>
<td>166</td>
<td>124</td>
<td>82</td>
</tr>
<tr>
<td>Water available</td>
<td>170</td>
<td>145</td>
<td>97</td>
<td>84</td>
</tr>
<tr>
<td>Organic matter</td>
<td>60</td>
<td>47</td>
<td>52</td>
<td>12</td>
</tr>
<tr>
<td>Slope</td>
<td>64</td>
<td>31</td>
<td>67</td>
<td>13</td>
</tr>
</tbody>
</table>

The controversy arises for the Pine Dominated Forest as in general case the steepness of the slope constitutes the resource for the other 3 forest types, since it forms the proper hydrological condition. However, for Pine Dominated Forest the situation arises just opposite meaning that flatness of terrain is a resource for some pine species. Therefore, the results would contain partially biased information about Pine Forest. Overcoming this problem requires more data manipulation going beyond the limits of this study.

Finally signatures were stored in separate input file with similar format as of one of ground facts input. However, there was still one problem that might result in research output. The order, in which signatures will take part in the analysis, plays significant role in results, e.g. when signature with lowest requirements take part first it will take all the possible area leaving no room for the remaining signature. Therefore, signatures were stored in decreasing order based on mean requirement value.
3.2. Flow of the Analysis

After preparation of input data and arrangement of algorithm procedures in orders proposed in two strategies the analysis was performed on 4 input bands with level of generalization 4 and edge width 1 pixel. Signatures were organized in a way, that lower bound of one land use (forest type) became upper bound of the next land use with lower level of requirements. The test program compiled from the code, gave the log, where the program was keeping track of changes and decisions made in final matrix.

For this particular case the Strategy 1 was used to simulate situation where forest types were distributed with both high and low performance groups: condition 1-3 (high) and condition 4-6 (low). As it was also pointed out in Chapter 2, Strategy 1 offers equal opportunities for each type of forest, so the performance would be some secondary objective (put in order of signatures) in the procedure. The main objective, here, was to distribute forest types just to fill out their natural condition niches. According to the model low performance area would decrease the effectiveness of land use for the whole area. However, this strategy would show natural patterns of forest distribution, what points out the ability of the model to be used not only for optimization.
Figure 3.6. Distribution of Forest Types within the Study Area According to Strategy 1.

As it is seen from the map above, Interpolation influenced the result of the analysis. In general, it placed the Forest Types with high level of requirements into
valleys where the depth of soil is higher and organic meter is available in higher volumes.

Also, it placed oak on the slope where it can find appropriate hydrological regime.

Figure 3.7. Distribution of Forest Types within the Study Area According to Strategy 2.
In case of Strategy 2 it is also noticeable that procedure placed the Forest Types in areas denoted in requirements. Even though the differences among strategies are almost unnoticeable visually at this extend, the over all areas of land uses differ in general by 100 pixels what is 3000 square meters (~0.8 acres) on the ground for each lend use.

**Table 3. Differences in Distribution of Forest Types between Strategies 1 & 2.**

<table>
<thead>
<tr>
<th>Forest Type</th>
<th>Forest Stand Condition Class</th>
<th>Priority</th>
<th>Strategy 1</th>
<th>Strategy 2</th>
<th>Differences</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Pixel</td>
<td>Tree</td>
<td>Pixel</td>
<td>Tree</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Count</td>
<td>Count</td>
<td>Count</td>
<td>Count</td>
</tr>
<tr>
<td>Maple Dominated Forest</td>
<td>1-3</td>
<td>1</td>
<td>15320</td>
<td>15494</td>
<td>30840</td>
</tr>
<tr>
<td></td>
<td>4-6</td>
<td>3</td>
<td>12361</td>
<td>12626</td>
<td>370830</td>
</tr>
<tr>
<td>Mixed Hardwoods</td>
<td>1-3</td>
<td>2</td>
<td>13831</td>
<td>13853</td>
<td>414930</td>
</tr>
<tr>
<td></td>
<td>4-6</td>
<td>5</td>
<td>50734</td>
<td>50810</td>
<td>1522020</td>
</tr>
<tr>
<td>Oak Dominated Forest</td>
<td>1-3</td>
<td>4</td>
<td>40788</td>
<td>41011</td>
<td>1223640</td>
</tr>
<tr>
<td></td>
<td>4-6</td>
<td>6</td>
<td>60275</td>
<td>60136</td>
<td>1808250</td>
</tr>
<tr>
<td>Pine Dominated Forest</td>
<td>1-3</td>
<td>7</td>
<td>610687</td>
<td>610072</td>
<td>18320610</td>
</tr>
<tr>
<td></td>
<td>4-6</td>
<td>8</td>
<td>149336</td>
<td>149216</td>
<td>4450140</td>
</tr>
<tr>
<td>Unassigned</td>
<td></td>
<td></td>
<td>46496</td>
<td>46532</td>
<td>1394890</td>
</tr>
</tbody>
</table>

Strategy 2 decreased the area occupied by lower priority forest types (Pine) while increasing the area of high priority Forest. This helped to redistribute resources to the benefit of high priority land uses, what in fact, increases the overall performance of the area.

Even though the Strategy 2 is performance oriented, the presence of lower condition class signatures brought inefficiency in the land use. To show the distribution of just highest condition class (high performance) Forest Types taking part in the analysis,
the data was processed once again with just four 1-3 Condition Class signatures. Doing this, an analyst excludes the ineffective land use and optimizes the forest growth in accordance to the model.

Figure 3.8. Optimized Distribution of Forest Types.
Table 4. Differences between Optimized and Un-optimized Distributions of Forest Types.

<table>
<thead>
<tr>
<th>Forest Type</th>
<th>Strategy 2 Un-optimized</th>
<th>Strategy 2 Optimized</th>
<th>Difference</th>
<th>Difference with Strategy 1 un-optimized</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pixel Count</td>
<td>Area</td>
<td>Tree Count</td>
<td>Pixel Count</td>
</tr>
<tr>
<td>Maple Dominated Forest</td>
<td>15494</td>
<td>464820</td>
<td>30988</td>
<td>15552</td>
</tr>
<tr>
<td>Mixed Hardwoods</td>
<td>13853</td>
<td>415590</td>
<td>27706</td>
<td>13901</td>
</tr>
<tr>
<td>Oak Dominated Forest</td>
<td>41011</td>
<td>1230330</td>
<td>82022</td>
<td>54620</td>
</tr>
<tr>
<td>Pine Dominated Forest</td>
<td>610072</td>
<td>18302160</td>
<td>1220144</td>
<td>722739</td>
</tr>
<tr>
<td>Unassigned</td>
<td>46532</td>
<td>1395960</td>
<td>195625</td>
<td>5868750</td>
</tr>
</tbody>
</table>

The procedure replaced low performance land uses by uses with higher performance and lower level of requirements. This allowed redefining the shapes (distribution) of land uses (Forest Types) in a manner that maximizes the performance not only for a single land use but also for the whole management area. The optimization brought more than 25 000 trees of higher quality for just oak alone and more than 220 000 for pine. From the point of view of a forester, these differences at extent of just 3 km² can make significant contribution in sustainability of the local forest planting business. Therefore, Strategy 2 of the analysis can become a valuable tool for preserving local economy.

The presence of low performance land use types in the first part of the analysis helped not only to simulate the real world situation, but also the signatures depicted the
areas that can be optimized. For a forest manager, it can bring the answer to the question of where the resources of the area are used inefficiently and where the land use should be changed. Also, the ability of the procedure to delineate the shapes of main land uses and to leave the rest of the area unassigned can help to find the places for introduction of other land uses without altering the productivity of main land uses.

Finally, from the point of view of nature conservation the figure 3.9 shows that the pine forest – even though some pine species are not native to the area (Gordon, R.B., 1966) – will perform better in terms of timber quality. Also, the maple dominated forest was not eliminated from the area even though it was introduced to this area in 19th century and consequently it is not an endemic specie for the natural hickory-oak local forest. This means that natural condition of the forest cannot be clearly measured by production-oriented indices. Therefore, the search for effective alternative should be continued in fields of forestry and nature preservation and detailed study of historical biodiversity change must be conducted before doing the analysis like this. In this research the algorithm does not help to restore all the natural condition of forest on the area (even though it may be possible in future); the main purpose here was to find the compromising Land Use Structure for the area that would be adapted to natural conditions as much as analyst can take it. This will prevent wasting the most valuable resource of all times – the land.
Conclusions and Discussion

The research constitutes the attempt to create a new approach to optimization in land use planning. By combining the methodologies of remote sensing and landscape ecology, it brings together multi-spectral analysis of digital imagery and analysis of landscape texture. These powerful techniques are used to classify and cluster the area of study to the best advantage that can be predicted in developed model, meaning that the procedure can help to configure or redistribute the area and resources among land use types in a manner that allows maximization of output, which can be received from utilization of the resources. In contrast to the traditional land use assessment and optimization techniques used by USDA and FAO, this methodology does not use linear optimization for individual map unit. When running the optimization, developed model uses the idea of common effort and possibility to bring together all the necessary resources from different map units that can help to achieve the goal of a particular land utilization type.

As a product of conducted research, the heuristic optimization model was developed, implemented and tested on the real world data. During this process the model revealed several disadvantages that are, basically, the result of novelty of the approach. Among, those disadvantages the most disturbing one is the absence of direct dependency between input and output as there is linear optimization (there is no function that would describe relationship between resources and expected benefit). Instead, the empirically derived levels of outputs and resources needed are used to analyze matrices taking into account mainly spatial aspect of optimization. This empirical backlash may also bring
some biases and diminish the objectiveness of the analysis, as it was with non-endemic maple dominated forest type. However, some more detailed study of native vegetation types would eliminate this disadvantage form the model.

Although the model have some disadvantages, the results of the applied procedures bring strong evidence that the land use can be optimized basing on natural characteristics of land and correctly stated requirements for land use development. In this particular case the land use structure on the area was optimized to hold higher performance land uses - Mixed Hardwood and Maple Dominated Forest of condition class 4-6 were replaced by Oak Dominated forest of condition 1-3 and also Pine Dominated Forest of classes 1-3 came on the places previously occupied by Oak Dominated Forest of condition class 4-6 (considered to be ineffective).

The results of the research also prove the ability of the model to solve land use optimization tasks effectively in terms of both analysis time and problem size (time and space complexity). Developed procedures present several advantages over previously developed procedures used in land evaluation studies, landscape ecology and remote sensing. As it was pointed out earlier, the model offers self placement and distribution for land uses with maximum adaptation to the natural patterns of resource spread on the area. Additionally, it provides not only the optimization land use intensity (as it does the FAO methodology), but also the optimization of land use extent.

Landscape ecology can use the model to delineate ecosystems and their functional elements as long as they bare some common property that can be expressed in some numerical value. This can improve the management of nature preservation objects. In
addition to that, the semi-lacunarity analysis together with edge search can help the landscape ecologist to deal more precisely with landscape texture. The Lacunarity Analysis used in landscape studies up to this time, developed by Dr Plotnik at al in 1993 can be used as an object for comparison, since it was recognized to give reasonable texture analysis in order to have proper grain size and extent of the studied phenomena.

First advantage is the consideration of landscape texture. Even though the Lacunarity Analysis does not reject the fact that a landscape can have different sizes of its texture elements, it helps just to determine the best unitary shape and size of grain (texture unit) for the analysis that would allow generalization with acceptable level of correspondence to the ground truth. In this case the texture of landscape is brought to one, static size and shape ignoring all the bigger or smaller texture elements. Instead, the semi-lacunarity analysis offers dynamic consideration of the landscape texture, catching its elements along the flow of the procedure, and giving far more precise dimensions of texture elements. All this gives an analyst the ability to conduct research on finer grain with bigger extent, what partially washes away the dilemma of study scale in landscape ecology.

Second advantage is the Edge Search, which goes even further in refining the shape of texture elements. It uses the results of semi-lacunarity analysis to capture the individual texture element and working with it individually, the procedure finds new area and resources that would support the function of this particular element. Therefore, functional neighboring treats these elements more like a vector defined map units rather than raster pixels, and in regard to this the procedure brings together two powerful tools
to analyze spatial patterns of the land properties. In addition, the procedure simulates the
dynamics of patch development focusing rather on functional land use units (communities) individually, than on land use on the extent area as a whole. This feature provides community oriented, dynamic approach for both landscape (ecosystem) development and land use optimization.

In the field of remote sensing, the procedure can also bring its advantages of systematic grouping of elements. It means that the product of phenomena detection on the ground can be given not only in form of individually detected area units (pixels), but also in form in functionally connected elements of the system formed by the phenomena. In addition, the procedure does not exclude the possibility of its modification and combination with other well-known and tested tools in remote sensing. For example, the semi-lacunarity analysis can be replaced by Nearest Neighbor algorithm and together with Edge Search modified for work with compactness indices it can provide a fast and valuable tool for detecting ground objects in relations to their shape.

Further development of ideas presented in research would lead to appearance of land use arithmetic. It would involve study of combinations of supplemental land uses and redefinition of signatures for newly derived land uses. It is worth studying, because some of the resources can be shared in available amounts without actual withdrawal of the resources (slope, aspect, …), while others can be used out in all available capacity (water, Na, Ca, …). Based on that, the fuzzy logic for land use can be implemented in the procedure. Also, the concept may be combined with neural networks that would help to advance the land use optimization even further. However, all these perspectives can only
become reality after crystallization of thoughts expressed in this study and after being proved on real world exemplas built on the basis of the developed model for land use optimization.

Even though there is still a tremendous gap in knowledge of the resource demands for different land uses and the distribution of those resources on the Earth surface, the analytical procedure developed in this research can become a valuable planning tool for finding a better use for scarce land resources. The model can also be used for management of risks and quality in agricultural production. Taking into account its performance characteristics, flexibility and feasibility for present level of technology; the procedure can occupy its place among modern raster analysis algorithms and bring its contribution to many fields of human activities that can use this data type for their development.
Bibliography.


Ohio Timber Price Report., July 14, 2003, Ohio Department of Natural Resources., Source: http://www.dnr.state.oh.us/forestry/Landownerasst/TPR0703.htm


#include <iostream>
#include <iomanip>
#include <fstream>
#include <cstring>
#include <string>
#include <vector>
using namespace std;
#include <cmath>
#include "matrix.h"
#include "signature.h"
#include "Pix.h"
#include "Wind.h"
#include "MatPix.h"
#include "patch.h"

bool string2int(const char*, int&);

string* split(string, string);

vector<vector<int>> conv_to_int(vector<string*>);

bool read_n_init(const char*, MTX&);

void find_core_area(MPX*, Signat, int, short);

void edge_search(MPX&, Signat, int);

void ask_for_signatures(vector<Signat>&);

int main(){
    string f_name; // name of the file
    //cin>>f_name;
    f_name="Input.txt"; //cin>>f_name;//get file name
    MTX Matrix;
    if(!read_n_init(f_name.c_str(), Matrix)){
        cout<<"\n file is NOT OK "<<endl;
        return 1;
    }
    cout<<"\n Bands’ file is OK\n"<<endl;
    MPX rmx; // matrix to work with
    rmx.mpx_f_mtx(Matrix);
    Matrix.M.clear(); //clear old
    vector<Signat> criteria; //actual vector for signatures to be
    //working with
    ask_for_signatures(criteria); //prompt user for
    cout<<"Enter your starting window size (level of
generalization):"; //hard code of widow size
    short wsize=0;
    //cin>>wsize; // prompt about widow size
    wsize=4; //hard code of widow size
cout<<"\n You entered :"<<wsize<<"\n Thank you! \n Now we will start"<<" the analysis ... \n"<<endl;
int lutN;
for(size_t pr=0;pr<criteria.size();++pr){//strategy - 1
lutN=pr+1;
MPX* rmxP; //pointer
rmxP=&rmx;
find_core_area(rmxP, criteria[pr], lutN, wsize);
rmx.show_Pstates();
}
for(size_t se=0;se<criteria.size();++se){
lutN=se+1;
edge_search(rmx, criteria[se], lutN);
rmx.show_Pstates();
} /*for(size_t pr=0;pr<criteria.size();++pr){//strategy 2
lutN=pr+1;
MPX* rmxP; //pointer
rmxP=&rmx;
find_core_area(rmxP, criteria[pr], lutN, wsize);
rmx.show_Pstates();
edge_search(rmx, criteria[pr], lutN);
rmx.show_Pstates();
}*/
cout<<endl;
cout<<" Final matrix"<<endl;
rmx.show_Pstates();
return 0;
}

//******************************************************************
//  Function:  ask_for_signatures(Signat& crteria)
//  Purpose:   prompts user for signature values
//  Parameters: criteria - array of signatures to be entered
//  Calls:      main
//******************************************************************
void ask_for_signatures(vector<Signat>& criteria)
{
  string f_name;   // name of the file
  //cin>>f_name;
  f_name="signatures.txt";       // cin>>f_name;//get file name
MTX Matrix;

  if(!read_n_init(f_name.c_str(), Matrix))
  {
    cout<<"\n file is NOT OK "<<endl;
    return ;
  }
  cout<<"\n Signatures FILE is OK"<<endl;
criteria.resize(Matrix.M.size());
for(size_t k=0;k<criteria.size();++k)
  {
    criteria[k].signature=Matrix.M[k][0];
  }
}
// Function: string2int
// Purpose: converts string into int
// Parameters: digit - char array to convert
// result - resulting int
// Calls: main. section of reading file
//------------------------------------------------------------------------------
bool string2int(const char* digit, int& result)
{
    result = 0;
    // convert each digit char and add into result.
    while (*digit >= '0' && *digit <= '9') {
        result = (result * 10) + (*digit - '0');
        digit++;
    }
    // check that there were no non-digits at end.
    if (*digit != 0) {
        return false;
    }
    return true;
}

//------------------------------------------------------------------------------
// Function: split
// Purpose: split a string into array of strings
// Parameters: target - string to split
// token - delimiter
// Calls: main. section of reading file
//------------------------------------------------------------------------------
string * split(string target, string token)
{
    int si=0;
    int start=0;   // string positions
    int end=0;   //
    string * vasa;   // new array
    int count=0;   // number of tokens in string
    if(target[0]== token[0]) {
        target.erase(0,1);
    }
    if(target[target.size()]==token[0]) {
        target.erase(target.size(),1);
    }
    while((start=(target.find(token,start)+1))) { // count the tokens
        count++;
    }
    start=0;
    vasa = new string[count+1];
    return vasa;
}
while(si<=count)
{
    end=target.find(token,start);
    vasa[si]=target.substr(start,(end-start));
    si++;
    start=end+1;
}
return vasa;
}

/**
 * Function: conv_to_int
 * Purpose: converts strings in matrix to integers
 * Parameters:
 *     inputM - matrix of strings
 * Calls:
 * main. section of reading file
 */
vector<vector<int> > conv_to_int(vector<string*> inputM)
{
    vector<vector <int> > target;
    size_t size=inputM.size();
    target.resize(size);
    vector<int> temp;
    temp.resize(size);
    char* pEnd;
    for(size_t i=0; i<size;++i)
    {
        size_t j=0;
        for(; j<size ;++j)
        {
            temp[j]=(int)strtod(inputM[i][j].c_str(), &pEnd);
        }
        target[i]=temp;
    }
    return target;
}

/**
 * Function: read_n_init
 * Purpose: read from file and init matrix
 * Parameters:
 *     f_name - name of the file
 *     Matrix - matrix to init;
 * Calls:
 * main section of reading file
 */
bool read_n_init(const char* f_name, MTX& Matrix){
    ifstream inFile2(f_name, ios::in);
    if (!inFile2){
        return false;
    }
    enum action {number, band, stop};
    action curr=number;
    string buffer;
int count=0;
int Bcount=0;
string token1=" ";
vector <string*> inputM;
int Bnumber=0;
int BsizeH=0;
int BsizeV=0;
vector<vector<vector<int> > > matrix;
while(!inFile2.eof()){
    getline(inFile2, buffer);
    if (buffer[0]=='
'){//skips the comment line
        continue;
    }
    switch (curr){
    case number:
        inputM.push_back(split(buffer,token1));
        if(!string2int(inputM[0][0].c_str(), Bnumber)){
            return false;
        }
        if(!string2int(inputM[0][1].c_str(), BsizeH)){
            return false;
        }
        if(!string2int(inputM[0][2].c_str(), BsizeV)){
            return false;
        }
        inputM.clear();
        curr=band;
        break;
    case band:
        inputM.push_back(split(buffer,token1));
        ++count;
        if(count==BsizeH){
            matrix.push_back(conv_to_int(inputM));
            count=0;
            ++Bcount;
            inputM.clear();
            if(Bcount==Bnumber){
                curr=stop;
            }
        }
        break;
    default:
        break;
    }
    continue;
}
inFile2.close();
if(curr!=stop){
    return false;
}
Matrix.set_mtx(matrix);
return true;
void find_core_area(MPX* rmx, Signat SinQ, int Cstate, short wsize)
{
    MPX dummy;
    Window wxy;
    cout<<"Widndow search for signature #"<<Cstate<<": ";
    for(size_t x=wsize;x>0;--x)     //||Window size
    {
        wxy.x=x;
        cout<<"x:";      //||
        for(size_t y=wsize;y>0;--y)    //||
        {
            wxy.y=y;
            for(size_t stx=0;stx+x-1<rmx->MP.size();++stx)
            //Start point
            {
                wxy.start_x=stx;       //manipulation
                for(size_t sty=0;sty+y-1<rmx->MP[0].size();++sty)
                {
                    wxy.start_y=sty;
                    if(rmx->give_window(dummy, wxy))
                    {
                        vector<int> cMeanV;
                        dummy.get_mean_vec(cMeanV);
                        Signat current(cMeanV);
                        if(SinQ<=current) //actual testing
                        {
                            //for suitability
                            rmx->change_state(dummy, Cstate);
                            cout<<"[]";
                        }
                    }
                }
            }
        }
    }
    cout<<endl;
}
void edge_search(MPX& rmx, Signat curLU, int lutN)
{
    Pix Starty;
    Patch landUseT;
    int patchC=0;     //patch counter
    while(rmx.give_start(Starty, lutN))
    {
        ++patchC;
        cout<<"\n For patch # "<<patchC<<", and landuse type 
 #"<<lutN<<endl;
        Starty.PatchN=patchC;
        landUseT.set_B_n_E(rmx, lutN, Starty);
        landUseT.show_B_n_E();
        rmx.update_patch_NS(landUseT.Body);
        while(landUseT.try_Edge(curLU.signature)){ // actual search in the edge
            rmx.update_patch_NS(landUseT.novi);
            landUseT.reset_B_n_E(rmx);
            rmx.update_patch_NS(landUseT.novi);
            landUseT.novi.clear();
            landUseT.show_B_n_E();
        }
        landUseT.Clear();
    }
}
/* end of test.cc*/
/* File: matrix.h */

#include <iostream>
using namespace std;

//******************************************************************
// Class: MTX
// Purpose: implements multy band Matrix
//******************************************************************
struct MTX {
    vector<vector<vector<int> > > M;
    MTX() {
    }
    void set_mtx(vector<vector<vector<int> > > src);~MTX() {
        M.clear();
    }
    void print_out();
};
void MTX::print_out()
{
    for(size_t i=0; i < M.size(); ++i)
    {
        cout << "BAND # " << i << endl;
        for(size_t j=0; j < M[i].size(); ++j)
        {
            for(size_t k=0; k < M[i][j].size(); ++k)
            {
                cout << M[i][j][k] << " ";
            }
            cout << endl;
        }
    }
    void MTX::set_mtx(vector<vector<vector<int> > > src){
        M = src;
    }
#endif /* end of matrix.h*/
/ File: Pix.h*/
#ifndef PIX_H
#define PIX_H
#include <vector>
#include <iostream>
using namespace std;

struct Pix{
    size_t x;
    size_t y;
    int state;
    vector <int> BVals;
    int PatchN;
    Pix()
    {
    
    }
    Pix(const Pix& source)
    {
        x=source.x;
        y=source.y;
        state=source.state;
        BVals=source.BVals;
        PatchN=source.PatchN;
    }
    ~Pix()
    {
        BVals.clear();
    }
    Pix operator=(Pix src)
    {
        x=src.x;
        y=src.y;
        state=src.state;
        BVals=src.BVals;
        PatchN=src.PatchN;
        return src;
    }
};
Pix Pix::operator=(Pix src)
{
    x=src.x;
    y=src.y;
    state=src.state;
    BVals=src.BVals;
    PatchN=src.PatchN;
    return src;
}
#endif
/* end of Pix.h*/
/* File: MatPix.h*/
#ifndef MPX_H
#define MPX_H
#include <vector>
#include <iostream>
#include "Pix.h"
#include "matrix.h"
#include "Wind.h"

using namespace std;

//******************************************************************
//  Structure:  MPX
//  Purpose: implements Matrix of Pix class
//******************************************************************
struct MPX{
  vector<vector<Pix > > MP;
  MPX()
  {
  }
  void mpx_f_mtx(MTX);
  void show_Pstates();
  bool give_window(MPX &, Window);
  void change_state(MPX &, int );
  void get_mean_vec(vector<int>&);
  bool give_start(Pix&, int );
  void give_edge_n_body(vector<Pix>&, vector<Pix>&, Pix );
  void update_patch_NS(vector<Pix>);
  void give_edge(vector<Pix>&, Pix);
  void update_Edge(vector<Pix>);
  ~MPX()
  {
    MP.clear();
  }
};

//*******************************************************************
//function : mpx_f_mtx(MTX)
//params   : MTX object to be converted
//*******************************************************************
void MPX::mpx_f_mtx(MTX src)
{
  MP.resize(src.M[0].size());
  for(size_t a=0; a< MP.size(); ++a)
  {
    MP[a].resize(src.M[0][0].size());
    for(size_t b=0; b<MP[a].size(); ++b)
    {
      MP[a][b].BVals.resize(src.M.size());
      for(size_t c=0; c<src.M.size(); ++c)
      {
        MP[a][b].BVals[c]=src.M[c][a][b];
      }
      MP[a][b].x=a;
      MP[a][b].y=b;
      MP[a][b].state=0;
      MP[a][b].PatchN=-1;
    }
  }
}
//function : show_Pstates()
//params : none
//Purpose : shows selected or unselected pixels in matrix of MPX object
//*******************************************************************
void MPX::show_Pstates()
{
    for(size_t a=0; a< MP.size(); ++a)
    {
        for(size_t b=0; b<MP[a].size(); ++b)
        {
            if(b!=0)
            {
                cout<<" "<<MP[a][b].state;
            }
            else
            {
                cout<<MP[a][b].state;
            }
        }
        cout<<endl;
    }
    cout<<"OK 
";
}

//function : show_PpatchN()
//params : none
//Purpose : shows patches in matrix of MPX object
//*******************************************************************
void MPX::show_PpatchN()
{
    for(size_t a=0; a< MP.size(); ++a)
    {
        for(size_t b=0; b<MP[a].size(); ++b)
        {
            cout<<" "<<MP[a][b].PatchN;
        }
        cout<<endl;
    }
    cout<<"OK 
";
}

//function : give_window(MPX & reff, Window w)
//params : reff - MPX object passed by reference, w - window with parameters for choosing pixels
//Purpose : dumps the pixels into reff to be processed in main
//*******************************************************************
bool MPX::give_window(MPX & reff, Window w)
{
    reff.MP.clear();
    MPX temp;
    temp.MP.resize(w.x);
for(size_t i=0; i<w.x;++i)
{
    temp.MP[i].resize(w.y);
}
for(size_t a=w.start_x; a<(w.x + w.start_x); ++a)
{
    for(size_t b=w.start_y; b<(w.y + w.start_y); ++b)
    {
        if(MP[a][b].state != 0)
        {
            return false;
        }
        temp.MP[a - w.start_x][b - w.start_y].BVals=MP[a][b].BVals;
        temp.MP[a - w.start_x][b - w.start_y].state=MP[a][b].state;
        temp.MP[a - w.start_x][b - w.start_y].x=MP[a][b].x;
        temp.MP[a - w.start_x][b - w.start_y].y=MP[a][b].y;
        temp.MP[a - w.start_x][b - w.start_y].PatchN=MP[a][b].PatchN;
    }
}
reff.MP=temp.MP;
return true;

//*******************************************************************
//function : change_state(MPX & adr, short State)
//params   :  adr - MPX object passed by reference(addresses for
//state to be changed)
//  State - state to what the Pixels’ states to be changed
//Purpose  : changes state of pixels in MP denoted in adr[][] .x and
//adr[][] .y
//  to the state passed as param
//*******************************************************************
void MPX::change_state(MPX & adr, int State )
{
    for(size_t i=0; i <adr.MP.size(); i++)
    {
        for(size_t j=0; j<adr.MP[i].size(); j++)
        {
            MP[adr.MP[i][j].x][adr.MP[i][j].y].state=State;
        }
    }
}

//*******************************************************************
//function : get_mean_vec(vector<int>& res)
//params    :  res - vector to be than converted to signature and
//processed
//Purpose : gives the mean values of each band of the MP in form
//of vector
//  correspondingly
//*******************************************************************
void MPX::get_mean_vec(vector<int>& res)
{
    res.clear();
    res.resize(MP[0][0].BVals.size());
    for(size_t v=0; v<res.size(); ++v)// initialization
    {
        res[v]=0;
    }
size_t count=0;
for(size_t i=0; i<MP.size(); ++i)
{
    for(size_t j=0; j<MP[i].size(); ++j)
    {
        ++count;
        for(size_t k=0; k<res.size(); ++k)
        {
            res[k]+=MP[i][j].BVals[k];
        }
    }
}
for(size_t v=0; v<res.size(); ++v)
{
    res[v]=res[v]/count;
}

//*******************************************************************
//function : give_edge(vector<Pix>& currE, Pix middle)
//params   :  currE - vector of pix that would be the edge for current pixel
//         middle - current pixel
//Purpose  : seeks for yet unocupied surrounding pixels (edge)
//*******************************************************************
void MPX::give_edge(vector<Pix>& currE, Pix middle)
{
    currE.clear();
    size_t start_x=middle.x;
    if(start_x>0)
    {
        --start_x;
    }
    size_t start_y=middle.y;
    if(start_y>0)
    {
        --start_y;
    }
    size_t end_x=middle.x;
    if(end_x<(MP.size()-1))
    {
        ++end_x;
    }
    size_t end_y=middle.y;
    if(end_y<(MP[0].size()-1))
    {
        ++end_y;
    }
    for(size_t i=start_x;i<=end_x;++i)
    {
        for(size_t j=start_y;j<=end_y;++j)
        {
            if(MP[i][j].state==0)
            {
                MP[i][j].state=-(middle.state);
                currE.push_back(MP[i][j]);
            }
        }
    }
}
```cpp
void MPX::give_edge_n_body(vector<Pix>& currE, vector<Pix>& currB, Pix middle) {
    currE.clear();
    currB.clear();
    size_t start_x = middle.x;
    if (start_x > 0) {
        --start_x;
    }
    size_t start_y = middle.y;
    if (start_y > 0) {
        --start_y;
    }
    size_t end_x = middle.x;
    if (end_x < (MP.size() - 1)) {
        ++end_x;
    }
    size_t end_y = middle.y;
    if (end_y < (MP[0].size() - 1)) {
        ++end_y;
    }
    for (size_t i = start_x; i <= end_x; ++i) {
        for (size_t j = start_y; j <= end_y; ++j) {
            if (MP[i][j].state == 0) {
                MP[i][j].state = -(middle.state);
                currE.push_back(MP[i][j]);
            }
            if (MP[i][j].state == middle.state &&
                (MP[i][j].PatchN != middle.PatchN)) {
                MP[i][j].PatchN = middle.PatchN;
                currB.push_back(MP[i][j]);
            }
        }
    }
}
```
bool MPX::give_start(Pix& Start, int lutN) {
    for(size_t h=0; h < MP.size(); ++h) {
        for(size_t v=0; v < MP[h].size(); ++v) {
            if(MP[h][v].state == lutN && MP[h][v].PatchN == -1) {
                Start = MP[h][v];
                return true;
            }
        }
    }
    return false;
}

void MPX::update_patch_NS(vector<Pix> body) {
    for(size_t h=0; h < body.size(); ++h) {
        MP[body[h].x][body[h].y] = body[h];
    }
}

void MPX::update_Edge(vector<Pix> edge) {
    for(size_t h=0; h < edge.size(); ++h) {
        MP[edge[h].x][edge[h].y].state = edge[h].state;
    }
}
#endif
/*end of MatPix.h*/
/*File: signature.h*/

#ifndef SIGNT_H
#define SIGNT_H
#include <vector>
using namespace std;

//****************************************************************************
//  Sruct:  SIGNATURE
//  Purpose: implements SIGNATURES for LUT
//****************************************************************************

struct Signat{
    vector<int> signature;

    Signat()
    {
        ;
    }
    Signat(int size, int value)
    {
        vector<int> src(size, value);
        signature=src;
    }
    Signat(const Signat& src)
    {
        signature=src.signature;
    }
    Signat(vector<int>src)
    {
        signature=src;
    }

    ~Signat()
    {
        signature.clear();
    }

    bool operator<=(Signat cmpr);
    Signat operator=(Signat src)
    {
        signature=src.signature;
        return src;
    }

    bool Signat::operator<=(Signat cmpr)
    {
        bool result=true;
        for(size_t i=0;i<signature.size();++i)
        {
            result= result && (signature[i]<=cmpr.signature[i]);
        }
        return result;
    }
};

Signat Signat::operator=(Signat src)
{
    signature=src.signature;
    return src;
}
// make comparison and gives improvement factor
//****************************************************************************

bool Signat::less_or_eq(Signat cmpr, double& impF)
{
    impF=0;
    bool result=true;
    for(size_t i=0;i<signature.size();++i)
    {
        result= result && (signature[i]<=cmpr.signature[i]);
        impF+=cmpr.signature[i]-signature[i];
    }
    impF=impF/signature.size();
    return result;
}
#endif
/* end of signature.h*/
/*File: Wind.h*/
#ifndef WIND_H
#define WIND_H

//******************************************************************
// Struct: Window
// Purpose: implements window for testing
//******************************************************************

struct Window{
    size_t x;
    size_t y;
    size_t start_x;
    size_t start_y;
};

#endif
/* end of Wind.h*/
/* File: patch.h*/
#ifndef PATCH_H
#define PATCH_H
#include <vector>
#include <list>
#include <iostream>
#include "Pix.h"
#include "MatPix.h"
#include "signature.h"

using namespace std;

//******************************************************************
//  Class:  LU
//  Purpose: implements Land Use object for the area
//******************************************************************

struct Patch{
    vector<Pix> Body;
    vector<Pix> Edge;
    vector<int> mean_Vs;
    vector<Pix> novi;
    Patch()
    {
    }
    bool set_B_n_E(MPX, int, Pix);
    void show_B_n_E();
    bool try_Edge(vector<int>);
    bool reset_B_n_E(MPX);
    ~Patch()
    {
        Body.clear();
        Edge.clear();
        mean_Vs.clear();
        novi.clear();
    }
    void Clear()
    {
        Body.clear();
        Edge.clear();
        mean_Vs.clear();
        novi.clear();
    }
};

void Patch::show_B_n_E()
{
    cout<<"\n Body ["<<Body.size()<<"] : ";
    /*for(size_t r=0;r < Body.size();++r)
    {cout<<Body[r].state<<", ";}
    */
    cout<<"\n Edge ["<<Edge.size()<<"] : ";
    /*for(size_t i=0;i < Edge.size();++i)
    {cout<<Edge[i].state<<", ";}
    */
    cout<<endl;
}
//*******************************************************************
//function : set_B_n_E(MPX source, int lutN, Pix first )
//params   :  source - Matrix to be examined, lutN - land Use
#(signature#),
//  first - starting pixel for the patch to be developed
//Purpose  : initializes the Body and Edge for the patch inside the
Matrix
//*******************************************************************
bool Patch::set_B_n_E(MPX source, int lutN, Pix first )
{
  int Npatch=first.PatchN;
  Body.clear();
  Edge.clear();
  source.MP[first.x][first.y].PatchN=Npatch;
  Body.push_back(first); //put in body and in
list<Pix> queue;
  queue.push_back(first); //temporary queue
  cout<<"Setting body : ... ";
  while(queue.size()! =0)
  {
    Pix current;
    vector<Pix> tempB;
    vector<Pix> tempE;
    current=queue.front();
    queue.pop_front();
    source.give_edge_n_body(tempE, tempB, current);
    for(size_t k=0;k<tempB.size();++k)
    {
      Body.push_back(tempB[k]);
      queue.push_back(tempB[k]);
    }
    for(size_t k=0;k<tempE.size();++k)
    {
      Edge.push_back(tempE[k]);
    }
  }
  cout<<endl;
  //source.show_Pstates();
  return true;
}
PUBLIC bool Patch::try_Edge(vector<int> Sign)
{
  int Npatch=first.PatchN;
  Body.clear();
  Edge.clear();
  source.MP[first.x][first.y].PatchN=Npatch;
  Body.push_back(first); //put in body and in
list<Pix> queue;
  queue.push_back(first); //temporary queue
  cout<<"Setting body : ... ";
  while(queue.size()! =0)
  {
    Pix current;
    vector<Pix> tempB;
    vector<Pix> tempE;
    current=queue.front();
    queue.pop_front();
    source.give_edge_n_body(tempE, tempB, current);
    for(size_t k=0;k<tempB.size();++k)
    {
      Body.push_back(tempB[k]);
      queue.push_back(tempB[k]);
    }
    for(size_t k=0;k<tempE.size();++k)
    {
      Edge.push_back(tempE[k]);
    }
  }
  cout<<endl;
  //source.show_Pstates();
  return true;
}
PUBLIC bool Patch::try_Edge(vector<int> Sign)
{
  //does not take care of parametr vector size
  bool result=false;
  if(Body.size()==0 || Body[0].BVals.size()!=Sign.size() )
  {
    return result;
  }
  //define the mean values of the patch
  if(mean_Vs.size()==0)
  {
    mean_Vs=Body[0].BVals;
    for(size_t i=1;i<Body.size();+i)
for(size_t j=0; j<Body[i].BVals.size(); ++j)
{
    mean_Vs[j]+=Body[i].BVals[j];
}

for(size_t h = 0; h <mean_Vs.size(); ++h)
{
    mean_Vs[h]=mean_Vs[h]/Body.size();
}

//*********************************defined
novi.clear();//clear the container for newly founded
size_t oper_count=1; //counts additions to body
int added=0;
cout<<"\n Searching in Edge : ";
while(Edge.size()==0 || oper_count > 0 )
{
    oper_count=0;
    size_t address=0;
    vector<int> temp_mean;
    double maxImF=0; // maximal improvement factor
    vector<int> cMax=Sign;//

    for(size_t k=0; k<Edge.size(); ++k)
    {
        double curImF=0; // current improvement factor
        temp_mean=mean_Vs;
        for(size_t j=0; j<Edge[k].BVals.size(); ++j)
        {
            temp_mean[j]=((temp_mean[j]*Body.size())+Edge[k].BVals[j])/(Body.size()+1);
        }
        Signat curMax(cMax);
        Signat curFact(temp_mean);
        if(curMax.less_or_eq(curFact, curImF))//compares to
        {
            if(maxImF<=curImF)
            {
                cMax=temp_mean;
                address=k;
                oper_count=1;
                maxImF=curImF;
            }
        }
    }
}

if(oper_count==1)
{
    Edge[address].state=Body[0].state;
    Edge[address].PatchN=Body[0].PatchN;
    Body.push_back(Edge[address]);//add to body
    novi.push_back(Edge[address]);
    Edge.erase(Edge.begin()+address);//and delete from
    edge;
    //replace mean_Vs with max
    mean_Vs=cMax;
    ++added;
    result=true;
}
} //we no longer need Edge in memory
    cout<<" found -: "<<added<<endl;
    return result;
}

//*******************************************************************
//function : reset_B_n_E(MPX source, short lutN)
//params   :  source - Matrix to be examined,
//Purpose  :REDEFINE the Body and Edge for the patch inside the Matrix
//*******************************************************************
bool Patch::reset_B_n_E(MPX source)
{
    //BODY IS ALREADY DENOTED IN SOURCE MATRIX BY PATCH#
    int lutN;
    lutN=Body[0].state;
    source.update_Edge(Edge);//set the edge to avoid repetitions
    list<Pix> queue;
    for(size_t a=0;a<novi.size();++)
    {
        //put new founded in
        queue.push_back(novi[a]); //temporary queue(FIFO)
    }
    novi.clear();
    cout<<"\n Reseting body ... ";
    while(queue.size()!=0)
    {
        Pix current;
        vector<Pix> tempB;
        vector<Pix> tempE;
        current=queue.front();
        queue.pop_front();
        source.give_edge_n_body(tempE, tempB, current);
        for(size_t k=0;k<tempB.size();++)
        {
            Body.push_back(tempB[k]);
            queue.push_back(tempB[k]);
            novi.push_back(tempB[k]);
        }
        for(size_t k=0;k<tempE.size();++)
        {
            Edge.push_back(tempE[k]);
        }
    }
    cout<<endl;
    return true;
}
#endif
/*end of patch.h*/