Formalization of Spatial Focal Operations on Air Pollution Using Function-based Spatio-temporal Layers and Algebraic Structures

Zahra Bahramian[1], Mahmoud Reza Delavar[2] , Mohammad Reza Malek[3] and Mir Abolfazl Mostafavi[4]
1: MSc. Student, GIS Division, Department of Surveying and Geomatic Eng., College of Eng., University of Tehran, North Kargar Ave., After Jalal Al Ahmad Crossing, 11155-4563, Tehran, Iran
E-mail: zbahramian@ut.ac.ir
2: Assist. Prof.,Center of Excellence in Geomatic Eng. and Disaster Management, Department of Surveying and Geomatic Eng., College of Eng., University of Tehran, North Kargar Ave., After Jalal Al Ahmad Crossing, 11155-4563, Tehran, Iran
E-mail: mdelavar@ut.ac.ir
3: Assist. Prof., Department of GIS, Faculty of Geodesy and Geomatics Eng., K.N.Toosi Univ. of Technology, Tehran, Iran
E-mail: malek@ncc.neda.net.ir
4: Assist. Prof., Dept. Of Geomatics Science, University of Laval, Quebec QC G1K 7P4, Canada
E-mail: Mir-Abolfazl.Mostafavi@scg.ulaval.ca

Abstract
At present, air pollution is one of the most important challenges in environmental sciences that is either directly or indirectly harmful for humans and the environment. Air pollution data Obtained from monitoring stations are point-wise and hence there are some information gaps between monitoring stations. Therefore, interpolation methods such as Inverse Distance weighting (IDW) are used in this paper for modeling between the monitoring stations. Function-based layer is defined as a new way to store these information that stores the function itself and not pixel value of raster data. Each function-based layer includes reference to the input data set and a list of parameters for this instance of the function. To analyze these data, values from a location and intermediate surroundings can be combined by focal operations.
In this research, air pollution monitoring stations are mostly static and the value of pollutants over the time is dynamic. So function-based layers have spatio-temporal components and the use of spatial analysis functions on spatio-temporal function-based layers is needed. Most of efforts in this area are appropriate only for special applications. On the other hand, spatial and spatio-temporal spaces are two spatial categories with the same internal structure. They only differ in their dimensions. In a general manner, the problem of handling time is then reduced to formalize static and dynamic (category) domain of analyses and find a functor to lift from static to dynamic domain. From a categorical viewpoint, primitive elements and basic operators have to be lifted to dynamic domain by functors. It will cause complex analyses to be lifted automatically. Functors are higher order functions that have function(s) as input and output. Haskell as a functional programming language is used for implementation of higher order functions.
In this paper, spatio-temporal data acquired from air quality monitoring stations distributed in Tehran, capital of Iran, is used. Here, existing focal operations are developed in a general manner. So, using category theory, focal operations lift from static to dynamic domain and spatial focal operations on spatio-temporal function-based layers are obtained with minimal new programming in Haskell. This paper outlines the results of implementation of lifting focal operations on spatio-temporal function-based layers. The implementation clarifies more the robustness and conciseness of the lifting approach. In the next step, we will extend the formalization to other operations on function-based layers, such as local and zonal operations.

Keywords: GIS, analysis, IDW, Function-based Layer, Focal operations, Spatio-temporal GIS, Category Theory, Functor, Functional Programming Language, Haskell, Environment, Air Pollution.

1 Introduction

Two major groups in geospatial abstractions are identified including field and object data models. Map algebra is a set of analytical techniques that have been widely adopted for raster-based GIS. Many applications of GIS utilize field data models. Some of these applications are in the area of spatial analysis that the similarity between the surface model and a well-defined mathematical function is clear. So, in these cases a new way to store surface information, function-based layer, is defined. Function-based layer is defined by its mathematical and spatial function and a number of bounded variables and refers to the input data set and a list of parameters for this instance of the function. So it is possible to manipulate these layers through calculations of functions, and manipulates Geospatial Information System (GIS) layers in a symbolic and formulaic form. Local,

focal and zonal functions can be applied on function-based layer. In this paper, function-based layer and the focal operation on it is implemented.
GISs are widely used to support spatially related decisions. On the other hand, time is inherently linked to geospatial concepts (Egenhofer and Mark 1995). Temporal data in GISs can be divided to temporal location (the position of objects change in time), temporal attribute (descriptive properties of objects change in time) and combination of them (both position and attributes of objects change in time). Here, we concentrate on objects that only change their attributes in time. Existing commercial GISs still have shortcomings in handling time. Therefore, existing analysis functions have to be lifted from static to dynamic. Most of the past efforts in this area have used computational techniques for special purposes and existing commercial GISs have only very limited support for it. Therefore, using a general concept to establish dynamic capabilities to a wide range of GIS analyses is useful. This is possible through formalizing static and dynamic domain of analyses and finding morphisms to lift from static to dynamic domain.
This paper is structured as follows: In section 2, field and object data models are described and the concept of Map Algebra is explained. Function-based layer and Map Calculus and their elements are introduced in section 3. Section 4 reviews the basic mathematical concepts of this work. In this section, algebraic structures, category theory, and functors are introduced. Section 5 contains the definition and specifications of a functional programming language that provides a framework for implementation of the proposed idea. The proposed methodology and case study undertaken are explained in section 6. Finally, section 7 provides conclusions and future works.

2 Map Algebra

There are two spatial data models in GIS including field data models that represent the geographical space as a continuous “field” or “surface”, and object data models which represent discrete entities in space. Field data models are based on the mathematical concept of a scalar field, which can be represented using Equation (1):

$$z = f(location) \qquad (\mathbf{1})$$

where *location* describes a set of coordinates. There is a function *f* that yields the value *z*. The function *f* is a mathematical function which operates on different locations.
The term "map algebra" was first introduced in the late 1970s and has since been used in loose reference to a set of conventions, capabilities, and analytical techniques that have been widely adopted for raster-based GIS. Map algebra is primarily oriented toward data that are static. In the original map algebra, each variable or "layer" is a bounded plane surface on which position is expressed in terms of Cartesian (X,Y) coordinates. The portion of that surface uniquely referenced by a given X,Y pair is termed a "location", and each location is associated with one recorded characteristics. Each of the distinct conditions depicted on a layer is termed a "zone", and each zone is represented by a numerical value.
Every map algebra operation accepts one or more of these layers as input and generates a single new layer as output. While individual operators are narrowly defined, the range of possibilities for combining these operations is entirely open ended. There are three major groups of operations including "local," "focal," and "zonal" functions (Tomlin, 1990). Local functions combine values from the same location. Focal functions combine values from a location and intermediate surroundings. Zonal functions combine values from all locations in a zone.

3 Function-based Layer and Map Calculus

Many GIS applications use field models. Some of these applications are in the area of spatial analysis that the similarity between the surface model and a well-defined mathematical function is the clearest. In these cases, function-based layers provide a new representation for the fields in GIS, based on the notion of spatial functions and their combinations (Haklay, 2004). It is defined by its mathematical and spatial function and a number of bounded variables, while the system takes care of representing and manipulating it in a way that is transparent to the user. Unlike current representations, GIS stores the function itself and not spatial objects which contain the *z* value in the Equation (1). The main strength of the new representation is the ability to treat each

analytical layer in its symbolic form, thus making it possible to manipulate GIS layers in a formulaic form. This can increase the range of the GIS analytical toolbox.
Following Tomlin's Map Algebra, the term "Map Calculus" is used for function-based representation. Map Calculus describes the application of function-based layers in a GIS. In Map Calculus, GIS layers are stored as functions, and new layers can be created by combinations of other functions. To understand how function-based layers works, it is best to look at an example: IDW is one of the spatial interpolation methods that operates on a set of sampled points ($L_1,L_2,\ldots,L_n$) and calculates the value for a new location *L'* by calculating (Equation 2):

$$L' = \frac{\sum_{i=1}^{n} \frac{1}{d_i^p} L_i}{\sum_{i=1}^{n} \frac{1}{d_i^p}} \qquad (\mathbf{2})$$

where d_i is the distance of *L'* to the location L_i, and *p* is a power of the distance. The search radius is taken as a parameter of the function. In a function-based layer implementation, GIS stores the template for IDW, the defined radius distance and a linkage to the set ($L_1,L_2,\ldots,L_n$). The process can be repeated for any set of points. More sophisticated interpolation treats in the same way. They might include references to the set of points that are being used as the source of spatial interpolation, as well as other parameters that are specific to each spatial function.
In Map Calculus, GIS layers are stored as functions, and new layers can be created by combinations of other functions. In Map Calculus, each function includes reference to the input data set, and a list of parameters for this instance of the function. Map Calculus provides an explicit representation of the spatial functions and provides a compact storage of function-based layer. In Map Calculus, the computation ends in the definition of the layer not with the production of the grid layer. By using Map Calculus, some of the problems associated with the raster-based approach are eliminated. In Map Calculus, there is no need to select a specific cell resolution or to consider the storage of large raster layers while excessive computations in Map Algebra might lead to operational compromises such as a decision to increase pixel size, which reduces the overall accuracy of the model. In Map Calculus, the final model includes all the building blocks and the system can understand the semantics of the model. This ability will allow the creation of an optimizer that reduces the number of calculations required to solve a specific problem. Map Calculus-enabled GIS is a GIS that has been enhanced with the required procedures and data structures to handle function-based layers.
To enable a GIS to handle function-based layers, it must be capable of creating functions from other functions, where some of the variables have been assigned while the rest will be evaluated at runtime. These capabilities exist in functional computer languages, and originate from Church's Lambda Calculus (Church, 1941).

4 Algebraic Structures, Category Theory and Functors

To formalize static and dynamic domain of analyses and finding morphisms to lift from static to dynamic domain, a more abstract view is needed that ignores those properties of operations which depend on the object they are applied to. The required abstraction is the subject of category. Category is one of algebraic operations.
An algebra describes an abstract class of objects and their behavior (Guttag and Horning, 1978). In other words, an algebra consists of a collection of types, operations, and axioms. Structure of operations in an algebra is independent of an implementation. Thus, behavior of many things, if their behavior is structurally equivalent, can be described with the same algebra. Algebraic structures seem operable in GIS applications to simplify thinking about them. For example, real world and our models of real world in GIS, representing a system that is part of reality, are in the same algebraic structures. This property enables us to construct a relationship between them in a special way to move from one space to another (Karimipour et al., 2005b).
Among many algebraic structures, categories are the most operable algebraic structure to GIS applications. "A category is a collection of primitive element types, a set of operations upon those

types, and an operator algebra that is capable of expressing the interaction between operators and elements" (Herring et al., 1990) (Figure 1).

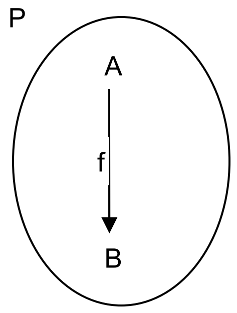

Figure 1. A Category with its elements.

In mathematical language, a category with its elements can be shown as Equation 3 (Karimipour et al., 2005b):

$$\forall A \in C, \exists e_A : A \to A, \ni [\forall f : A \to B, g : B \to A \Rightarrow e_A . f = f, g.e_A = g]$$
$$\forall f : A \to b, g : B \to C, \exists h : A \to C, \ni h = f.g \qquad (\mathbf{3})$$
$$\forall f : A \to B, g : B \to C, h; A \to C \Rightarrow (f.g).h = f.(g.h) = f.g.h$$

A categorical viewpoint demonstrates that semantics of operations are independent of the representations they are applied to (Frank, 2005). Category theory gives us a very high level abstract viewpoint: we directly address the properties of the operations, instead of discussing the properties of individual objects. This corresponds to the interest in geography, where the discussion concentrates on processes that occur in space, not on the collection of locations and properties of spatial objects (Frank, 2005). Because of these properties, we enable to have a function between two categories with the same internal structure as "functor".
A functor, or more properly morphism, is the process that associates elements and operations from one category to another that preserves the operator algebra (Herring et al., 1990). Such relations are commonplace in GIS. For example real world and a defined model of it, can be considered as two categories with the same structure and, therefore, it is possible to define a functor to associate them together. In GIS, such mappings that preserve algebraic structure are well-known as homomorphism (Karimipour et al., 2005b).

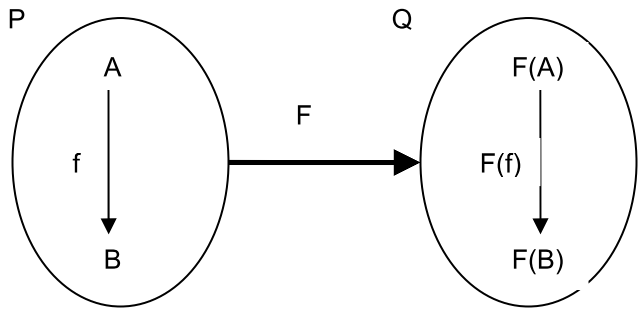

Figure 2 . Functor F transforms the first category elements to their associated one

Regarding the concept of categories, a morphism or more properly a functor between two categories two categories are considered named *P* and *Q*, Respectively. The equation 4 is confirmed:

$$\begin{gathered}F(e_A)=e_{F(A)}\\ \forall f:A\to B\in P, g:B\to C\in P\Rightarrow \\ [F(f):F(A)\to F(B)\in Q, F(g):F(B)\to F(C)\in Q, \ni F(f.g)=F(f).F(g)]\end{gathered} \qquad (4)$$

Lifting primitive elements and basic (non-decomposable) operators to dynamic domain will cause all analyses (combination of basic operators) to be lifted automatically.
On the other hand, static and dynamic models in GIS have the same internal structure. They represent the same concept in the real world and the concept of IDW interpolation in spatial and spatio-temporal spaces is the same. The only difference in these two spaces is in the dimension of their objects. Since category theory concentrates on the processes instead of properties of objects, it seems an appropriate candidate to associate spatial and spatio-temporal spaces and their objects and processes (Karimipour et al., 2005b). To lift (spatial operations) from spatial category to spatio-temporal category, we need functor. From this point of view, the time lifting is a functor from a static category to a dynamic one (Karimipour et al., 2005a).

5 Higher Order (Functional) Programming Languages

Programming languages are classified by orders that are the type of their variable symbols. A zero order language has no variables (only constant). A first order language has variables, which stand for objects, but not for functions. A second order language has variables that can stand for objects or functions (Frank, 2005b). Most previous research in spatial information theory is carried out using first order languages. However, some of the new applications need a higher order language (Frank, 1997).
Functional programming, is one of a number of different programming styles or paradigms; others include object-oriented (OO), structured and logic programming. It is a higher order programming language and concentrates on the relationships between values. The function is the fundamental building block of a functional programming language in which everything is a function (Bahramian et al., 2008).
Higher order functions allow to separate the part of operations specific to the data structure from the code of the operations which is specific to the data type stored. GIS are large data collections and must use complex spatial data structures. It is beneficial to separate the code which traverses the data structure from the code which operates on the feature data (Frank, 1997).
On the other hand, functors are higher order functions and have other functions as argument. So, the proposed treatment for dynamic construction of static analyses of GIS also needs a higher order programming environment. This is the case because functors can operate on any object that is a function (Karimipour et al., 2005b).
Haskell is one of the functional programming languages that support our requirements (Haskell website). It has evolved significantly since its birth in 1987. Here, Haskell is used for implementation. More details about Haskell can be found in Thompson (1999) and Haskell website.

6 Methodology and Case Study

Air pollution is one of the important problems we face in environmental and natural resource management. Air pollution is defined as any atmospheric condition in which certain substances present in such concentrations and duration that they may produce harmful effects on man and his environment (Rahmatizadeh, 2005). There are pollutants such as CO, NO_2, SO_2 and PM_{10} which are measured in air pollution stations.
Air pollution data is related to a specific location. GIS can be used to handle these spatially related data. There are some approaches that model the distribution of specific pollutant according to pollution source using mathematical, physical and chemical rules. These approaches are based on pollution sources not measured value of pollutant in pollution stations. Since there are some unknown pollution sources, these approaches are not appropriate for wide area. In addition, in these approaches, complete physical and chemical parameters are needed that are impossible. On the other hand, interpolation model only use the measured values for

pollutants in pollution stations and so is independent of pollution source. Thus, in this case, an interpolation model is a useful method. Interpolation models are used to model air pollution between air pollution monitoring stations in different location. Based on theoretical studies among the various interpolation methods, Kriging and IDW provide the perfect results (Rahmatizadeh, 2005). In this paper, IDW function-based layer is implemented and the operations are applied on them. Other interpolation methods can be used in a similar way.
In this paper, IDW interpolation is used to model air pollution between air pollution monitoring stations in different location. Our category has the following content:

• Primitive elements: Function-Based Layer
• Basic operators: +, * and so on.
• Functions and analyses: Local, Focal, Zonal operations and more complex operations

First, we construct function-based layer using IDW interpolation in Haskell. In Map Calculus-enabled GIS, when the user requests the GIS to calculate the function, the system registers the manipulation in a symbolic form. The results are illustrated in Figure 3. Apart of the functional programming undertaken in this paper is as follow:

```
idw :: (Float,Float) -> Float                          (5)
idwFunctionBasedLayer :: Layer Float
```

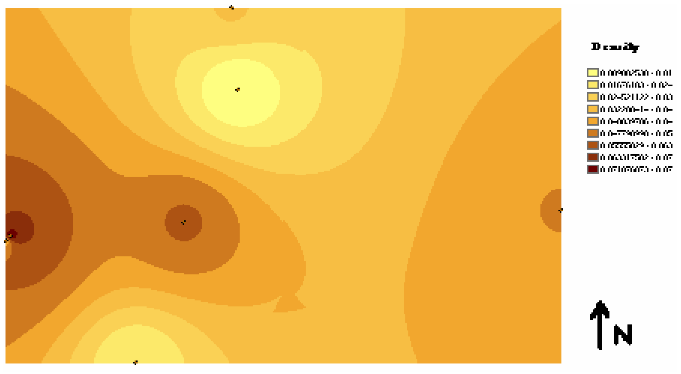

Figure 3.IDW Function-Based Layer for NO2 in March 29, 2004.

The operations on function based layers can be grouped to three basic operations: local, focal and zonal operation. Complex operations can be defined by composition of these basic operations. The next stage is applying focal operation on the function-based layer. In this paper, focal operation on function-based layer is implemented using in Haskell (Equation 6).

```
apply_focal_op :: Num a => ([a] -> a) -> Layer a -> a -> Layer a -> Layer a(6)
```

Time is defined as an Integer number (Equation 7).

```
type Time = Integer                                                        (7)
```

To lift the primitive elements, we define type “Changing” that changes each constant input *v* to a function from a Integer number to *v* (Equation 8):

type Changing v = Time -> v (8)

To lift the basic operators, such as +, a functor is needed. We define functor lift1 to lift operators with one Parameter (Figure 4). This functor add a parameter “t” to input functions (Equation 9).

```
lift1 :: (b -> c) -> a b -> a c                                            (9)
instance Lifts ((->) Time) where
  lift1 op a = \t -> op (a t)
```

Lifting for operators with more arguments can be done in a similar way.
The result of sum focal operation on IDW Function-Based Layer for March 29, 2004 is shown in Figure 4. We first obtain the result based on t and then substitute the value of t in it.

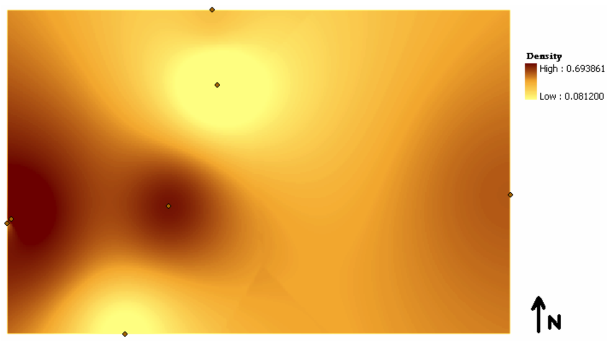

Figure 4. Focal operation (sum) on IDW Function-Based Layer for NO2 in March 29, 2004.

7 Conclusion and Future Works

This paper outlines the results of implementation of lifting focal operations on spatio-temporal function-based layers. Here, morphism was defined between static and dynamic spatial focal analyses of function-based layers to lift static spatial focal analyses to their temporal ones with less change.
In this paper, Haskell was used for implementation of the proposed method. Because Haskell is a functional language programming, written codes are shorter than their similar codes in other programming environments and they are more comprehensible and transparent, too. In addition, the implementation of functors has been achieved functional programming language. Time is included in GIS analysis with less change in static version of them and static and dynamic focal operations have been integrated into a unique algorithm.
Using category theory and functional programming language, the algorithm has been implemented effectively. The implementation of the morphism in a functional programming environment clarifies more the robustness and conciseness of the lifting approach. The results certify validity of these approaches for lifting focal operations on spatio-temporal function-based layers.
Applying more complicated analysis and also integration of these dynamic analyses with other applications that needs dynamic analyses as prerequisite are considered for future works. In the next step, we will extend the formalization to other operations on function-based layers, such as local and zonal operations.

References

1- Bahramian, Z. and M. R. Delavar, 2008. A comparative study of map calculus and map algebra in an GIS environment, *Proc., ACRS 2008 Conference., SriLanka, Nov. 10-14, 2008*.

2. Church, A., 1941. The Calculi of Lambda Conversion (*Princeton: Princeton University Press*).

3. Egenhofer, M.J., and D.M. Mark, 1995. *Naïve Geography*, National Center for Geographic Information and Analysis.

4. Frank, A.U. 1997. "Higher order functions necessary for spatial theory Development ". *Proceedings of Auto-Carto 13, Vol. 5, ACSM/ASPRS*, Seattle, Washington, April 7-10, 1997, pp. 11-22

5. Frank, A.U. (2005a). Map Algebra Extended with Functors for Temporal Data. *Proceedings of the ER Workshop 2005 (CoMoGIS'05)*. J. Akoka et al. Klagenfurt, Austria, Springer-Verlag Berlin Heidelberg. LNCS .

6. Frank, A. U.,2005b, *A Theory for Geographic Information Systems*, 2005.

7. Guttag, J. V. and Horning J. J., *The algebraic specification of abstract data types, Acta Informatica* 10(1): 27-52, 1978.

8. Haklay, M. ,2004. ‘Map Calculus in GIS: a proposal and demonstration’, *International Journal of Geographical Information Science*, Vol. 18**,** No. 1, pp. 107-125.

9. Haklay, M., 2007. Comparing Map Calculus and Map Algebra in Dynamic GIS, in Drummond, J., Billen, R., Forrest, D. and João, E. (eds), *Dynamic & Mobile GIS: Investigating Change in Space and Time .* Taylor & Francis.

10. Haskell website: https://www.haskell.org.

11. Herring, J., M. J. Egenhofer and A. U. Frank (1990). Using Category Theory to Model GIS Applications, Proc., *4th International Symposium on Spatial Data Handling*, Zurich, Switzerland.

12. Karimipour, F., M. R. Delavar, A. U. Frank and H. Rezayan (2005 a). Point in Polygon Analysis for Moving Objects, Proc., *DMGIS Confe*rence, Wales, U.K.

13. Karimipour, F., M. R. Delavar, A. U. Frank (2005 b). Applications of category theory for dynamic GIS analyses, Proc., *GIS Planet Conference*, Portugal.

14. Karimipour, F. (2005 c). Logical Formalization of Spatial Analyses of Moving Objects Using Algebraic Structures, M.Sc. Thesis, (in Persian with English abstract), College of Eng., University of Tehran, Tehran, Iran.

15. Karmipour, F., Delavar, M.R., Rezayan, H.: Formalization of moving objects’ spatial analysis using algebraic structures. Proc. Conf., *Extended Abstracts of GIScience 2006 Munster, Germany*, IfGI Prints, Vol. 28, pp. 105–111

16. Medak, D., 2003. Haskell Toturial, A Note for Learning Haskell, Institute for Geoinformation, Technical University of Vienna, Austria.

17. Rahmatizadeh, Sh., 2005, Design and Implementation of a Customized GIS for Air Quality Management. MSc. Thesis (in Persian with English abstract), Faculty of Geodesy and Geomatic Eng., University of KNT., Tehran,Iran.

18. Thompson, S., 1999. Haskell, The Craft of Functional Programming, Addison-Wesley Longman.

19. Tomlin, D., 1990. Geographic Information Systems and Cartographic Modeling. Englewood Cliffs: Prentice Hall.