Visualization of Recursively Defined Concepts *

Peter Revesz Shasha Wu
Dept. of Computer Science, Univ. of Nebraska-Lincoln, Lincoln, NE 68588
{*revesz, shwu*}*@cse.unl.edu*

Abstract

Visualization of recursively defined spatio-temporal concepts is a general problem that appears in many areas. For example, drought areas based on the Standardized Precipitation Index (SPI) and long-term air pollution areas based on safe and critical level standards are recursively defined concepts.

In this paper, we develop a general and efficient representation and visualization method for recursively defined spatio-temporal concepts. We illustrate our general method by visualizing drought and pollution areas.

1. Introduction

In Geographic Information Systems [4] we frequently need to visualize on a map the area where a given property P holds. The area is defined using the following *non-recursive* form.

Definition 1.1 An area has property P during time unit T, if during T we measure an amount k or more of an indicator of property P.

However, many properties cannot be defined in this simple way. Usually, these complex properties are defined based on a series of observations in time. Their definitions have the following general *recursive* form.

Definition 1.2 An area has property P during time unit T if during T we measure either
(i) k or more amount of an indicator of property P or
(ii) between k_1 and k amount of the same indicator
and the area has property P during time unit $T-1$.

The amount k_1 is less than k in the above definition. The first part of the recursive definition is like the Definition 1.1. The second part adds more areas. Hence while an area with only a measurement value between k_1 and k during time T does not have the property P according to the non-recursive definition, it *may* have the property according to the recursive definition. Definition 1.2 is appropriate when the indicators measure at time unit $T-1$ do not disappear completely by time unit T.

Example 1.1 Suppose we would like to find the counties on a map that have a *disease out-of-control* at time T, and suppose the only indicator that we have available is the number of new infections. Answering this query is not a simple matter of finding the county that have more than some k(e.g., $k = 10$) new infected persons. Some people infected with the disease at time $T-1$ will continue to be infected at time T. Suppose we expect about half of the infected persons to continue to be infected after a time unit. If at time $T-1$ a county had a disease out-of-control, then it is reasonable to assume that the disease is still out-of-control at time T if during T we have between five and nine new infected persons. Hence the recursive definition, written in the form of Definition 1.2, is as follows:

A county has *a disease out-of-control* during week T if during week T it reports either
(i) 10 or more new infected persons or
(ii) between 5 and 9 new infected persons and it is highly-infected during week $T-1$.

In this paper we describe an Information Visualization System for Recursively Defined Concepts (IVSRDC) that is able to visualize all recursively defined concepts expressible by Definition 1.2. Revesz and Li provided constraint-based visualization for spatio-temporal data in [7] but did not consider recursively defined concepts.

The paper is organized as follows:

Section 2 describes the main functions of the visualization system from the user's perspective. Section 3 describes the implementation of the system. Section 4 describes two sample applications (drought analysis and air-pollution evaluation) and reports the performance on the drought analysis problem. Finally, Section 5 concludes this paper and gives some directions for further work.

* This research was supported in part by USA NSF grant EIA-0091530. The first author was also supported by a Gallup Professorship.

2. The outline of the IVSRDC system

The Information Visualization System for Recursively Defined Concepts (IVSRDC) is a web-based general solution for those problems that can be defined by Definition 1.2. Figure 1 shows the abstraction of the system. There are four inputs for the system. One is the relation $M(x, y, t, w)$, where the attributes (x, y) specify 2-D locations, t specifies a time instance, and the last attributes w records the measurement of the indicator of property P of each location. The other three inputs are T, k, and k_1, which are defined in Definition 1.2. The output of the system is the visualized image(s) of a spatio-temporal relation $P(x, y, t)$ that represents the area with property P at time t.

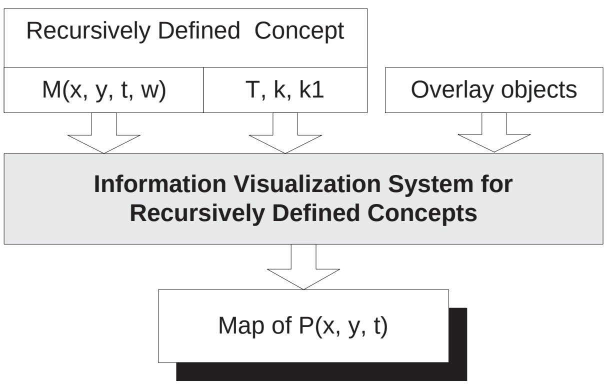

Figure 1. The functionalities of the IVSRDC.

The homepage of the system is shown in Figure 2. It provides an English description of the general problem that this system can solve and several input areas for the users to specify arguments related to their own applications.

The general solution for the problem defined in Definition 1.2 can be formally expressed as follows:

Given $M(x, y, t, w)$, we have:

$$A = \{(x, y, t) \ | \ M(x, y, t, w) \wedge w \geq k \wedge t \leq T\}$$
$$B = \{(x, y, t) \ | \ M(x, y, t, w) \wedge k_1 < w < k \wedge t \leq T\}$$

Where A is the part of M that is greater or equal to k, and B is the part that is between k and k_1. We define the areas having property P at time t after the Definition 1.2 as follows:

Definition 2.1
$$P = \{(x, y, t) \mid A(x, y, t) \ \vee (B(x, y, t) \wedge P(x, y, t-1))\}$$

Applying a general solution based on the general definition makes the system have the ability of solving similar problems without modifying the program. The user does not need to search for solutions in many specific applications. Our experience shows that this implementation largely increases the usability and maintainability of this system in practice. As we explain in Section 4, the applications of drought analysis and air-pollution evaluation have distinguish contexts. However, both of them can be described by the general definition of Definition 1.2. That enables us to solve them with the same system without changing the program.

3. Implementation of the IVSRDC system

In this section, we introduce some background information first. Then we describe the software architecture of the system and the data translation process. Finally, we illustrate the naive implementation and the optimized algorithm.

3.1. Background information

In a 2-D spatial problem, a point-based spatio-temporal relation has the schema of (x, y, t, w_1, w_2, ..., w_m), where the attributes (x, y) specify point locations, t specifies a time instance, and w_i $(1 \leq i \leq m)$ records the features of each location.

A point-based spatio-temporal data set only stores information of some sample points. To represent the features beyond those finite sample points, it is necessary to do spatio-temporal interpolation on them. In this paper, we use a 2-D spatial interpolation function for triangles [2, 3], which interpolates and translates the original point-based spatio-temporal information into a constraint relation.

A constraint database is a finite set of constraint relations. A constraint relation is a finite set of constraint tuples, where each constraint tuple is a conjunction of atomic constraints using the same set of attribute variables [6]. Hence, constraints are hidden inside the constraint tables, and the users only need to understand the logical meaning of the constraint tables as an infinite set of constant tuples represented by the finite set of constraint tuples. Typical constraints include linear or polynomial arithmetic constraints.

Management of Linear Programming Queries (MLPQ) system is a constraint database prototype system that implements rational linear constraint databases and queries. It supports both SQL and Datalog queries, minimum and maximum aggregation operators over linear objective functions among other functionalities [6].

By using constraint databases, even simple query languages, such as SQL and Datalog, can express some difficult recursively defined spatio-temporal concepts. With the help of the MLPQ system, the result of those queries can be automatically displayed in static snapshots or animation.

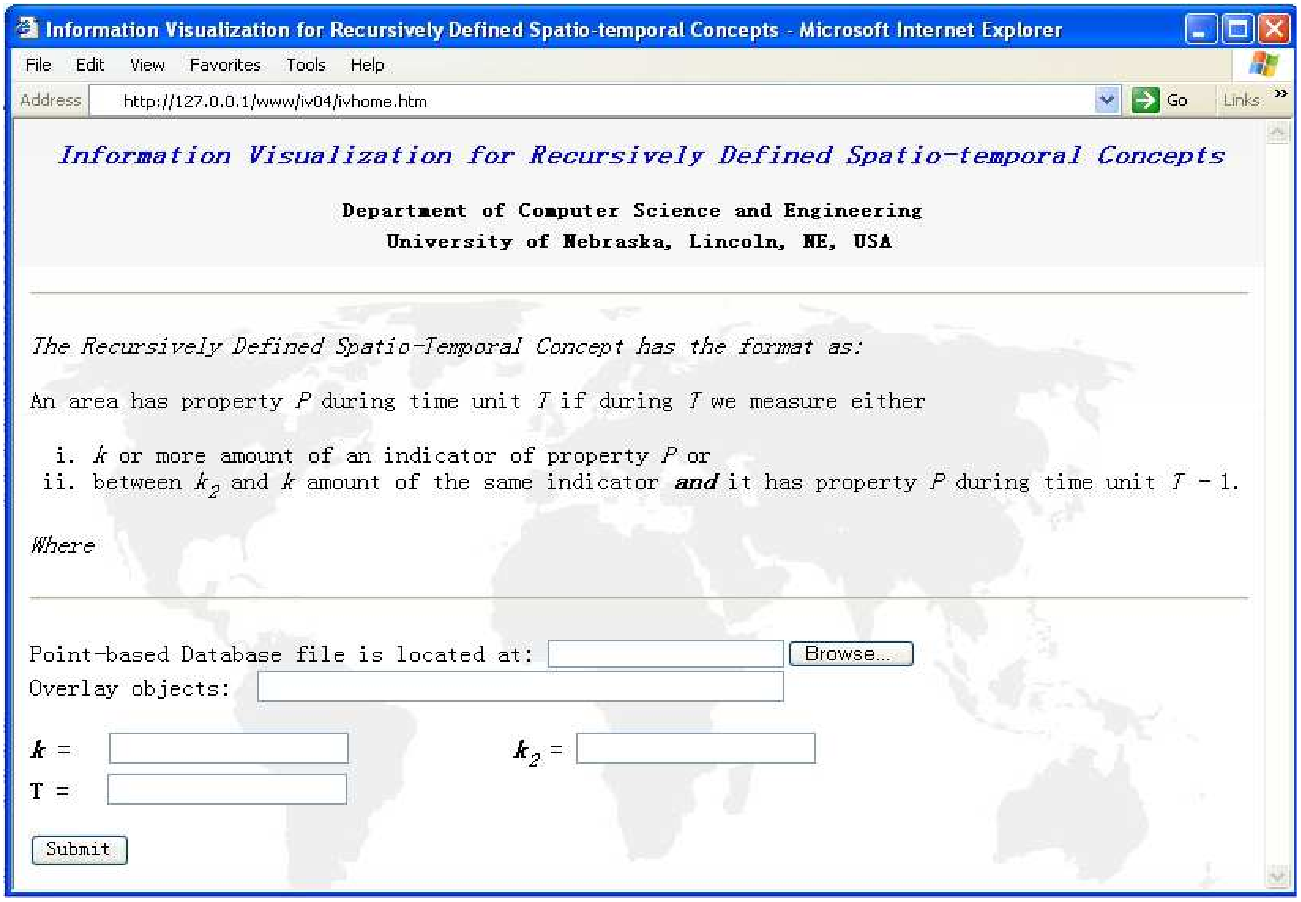

Figure 2. The homepage of the IVSRDC system.

3.2. Software architecture of the IVSRDC

Recursively defined spatial-temporal information is difficult to visualize in most systems. However, constraint databases [6] provide an efficient way to store the spatio-temporal data, and Datalog query language supports recursion. In this paper, we combine 2-D interpolation function and recursive Datalog with MLPQ constraint database system. Furthermore, we realize a general solution to calculate and visualize the complex spatial-temporal problems formulated according to Definition 1.2.

Figure 3 shows the software architecture of the IVSRDC system. It has three layers, which are described below.

I. User interfaces layer accepts input from the user and displays the output to the user. There are two inputs from the user. First, the point-based spatio-temporal data set. Second, the three arguments k, k_1, and time T.

II. Middle layer does the internal data processing. It has the following three modules.

1. The *interpolation module* is used to interpolate and translate point-based relational data into constraint data, It implements the 2-D shape function for triangles [3] to interpolate and translate the original data set into a constraint data set.

For example, Figure 4 is a point-based spatio-temporal data set consisting of Standardized Precipitation Index (SPI) records collected at 48 main weather stations spread out all over Nebraska.

The translation has two steps. First is a triangulation of the sample points. Several efficient algorithms have been developed to generate triangular meshes. A popular method among them is the "Delaunay Triangulation" [1, 9]. We embed in our system the Delaunay triangulation algorithm available from the public website `www.geom.umn.edu/software/~qhull`. Figure 5 shows the triangulation result of the 48 weather stations in Nebraska. Each of these stations is an extreme point of at least one triangle in the map.

The second step consists in defining for each triangle a linear interpolation function that represents the amount of indicator for any point within it [3]. We implement the algorithm in our system as a function of

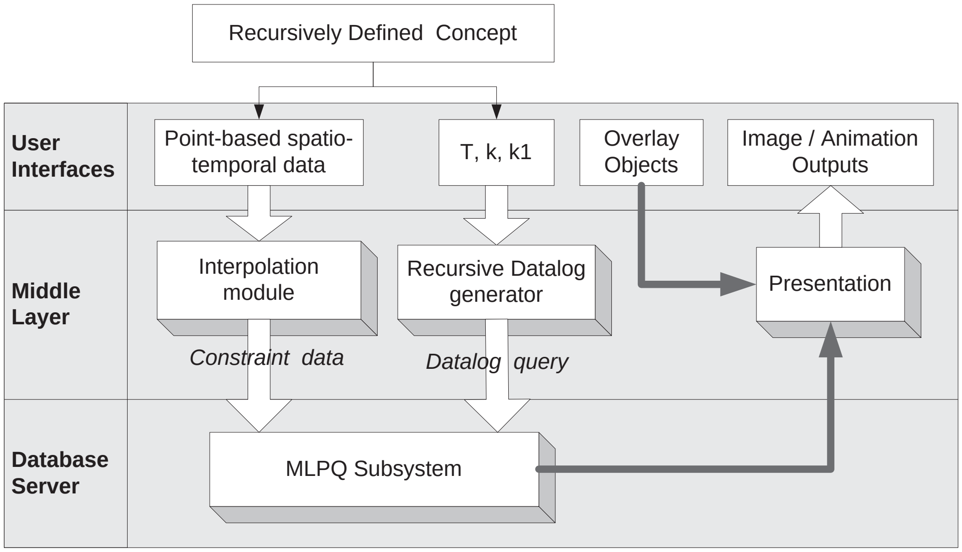

Figure 3. The software architecture of the IVSRDC system.

station	x	y	year-week	SPI
butte	-231.4	2214.9	2001-1	-0.62
bloomfield	-134.6	2179.1	2001-1	-0.83
oneill	-214.9	2164.1	2001-1	-0.83
⋮	⋮	⋮	⋮	⋮
butte	-231.4	2214.9	2001-12	-0.14
bloomfield	-134.6	2179.1	2001-12	0.07
oneill	-214.9	2164.1	2001-12	-0.15
⋮	⋮	⋮	⋮	⋮
butte	-231.4	2214.9	2002-1	-0.38
⋮	⋮	⋮	⋮	⋮

Figure 4. The point-based weekly SPI data.

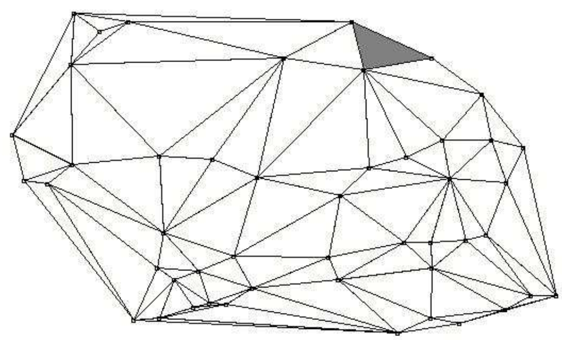

Figure 5. Triangulated map based on the 48 weather stations in Nebraska.

the *2-D interpolation module*.

Figure 6 shows the result of the interpolation and translation. It is a constraint relation with three constraint tuples. Each constraint tuple contains four linear constraints. The first three inequality constraints over x and y represent the triangular area. The last linear equation over x, y, and w represents the SPI value of the point at location (x, y) within this area. The field $week$ represents the weekly time starting from January 1st, 1800.

For example, the first tuple with $id = 1$ in Figure 6 is interpolated from the first three weather stations of Figure 4 and represented as the gray region in Figure 5. The same triangular region in Figure 5 has different tuples at different time instance in Figure 6. The time unit $week = 10489$ in Figure 6 corresponds to the first week of 2001 in Figure 4, $week = 10500$ represents the 12th week of 2001, and so on. The only difference between these tuples is the last linear equation over x, y, and w.

2. The *recursive Datalog Generator* is designed to generate the Datalog query with the input arguments and send the query to the MLPQ constraint database subsystem.

3. The *presentation module* explains and visualizes the output of the database layer. It also allows user to zoom in and zoom out any specific area within the whole map. It can also overlay some input objects on top of the output and generate a combined output image. Those objects may include state/county boundaries, highways or rivers.

id	east	north	week	SPI	
1	x	y	10489	w	$3.0788x + y \geq 1502.47,$ $-0.1868x + y \geq 2204.24,$ $0.3698x + y \leq 2129.32,$ $0.7280x - 3.8974y + 1000w = -9420.80.$
2	x	y	10489	w	$-5.6429x + y \geq 5432.03,$ $-1.4370x + y \leq 3040.58,$ $-0.9106x + y \geq 2694.89,$ $12.6524x + 0.8319y + 1000w = -6314.32.$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
1	x	y	10500	w	$3.0788x + y \geq 1502.47,$ $-0.1868x + y \geq 2204.24,$ $0.3698x + y \leq 2129.32,$ $-2.5483x - 1.0246y + 1000w = -1819.61.$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Figure 6. The constraint-based weekly SPI data set.

III. Database layer: The *MLPQ constraint database subsystem* is used to evaluate the input Datalog queries and visualize the result as animation or image(s). Finally, the *presentation layer* generates a new web page with result image(s) and display it to the users.

3.3. Naive algorithm and optimization

Definition 2.1 can be translated into the following Datalog *Query 1*:

$$
\begin{array}{lll}
A(x,y,t) & :- & M(x,y,t,w),\ \ w \geq k,\ \ t \leq T. \\
B(x,y,t) & :- & M(x,y,t,w),\ \ k_1 < w < k,\ \ t \leq T. \\
 & & \\
P(x,y,t) & :- & A(x,y,t). \\
P(x,y,t) & :- & B(x,y,t),\ \ P(x,y,t-1).
\end{array}
$$

Datalog *Query 1* is the core program of the IVSRDC system and can be evaluated by the MLPQ subsystem directly. Comparing the size and complexity of a C++ or Java program needed to solve the same problem based on relational databases, the Datalog and constraint databases provide a more concise and manageable approach. A simple and independent query solution makes the program easy to understand and maintain.

Although the code of naive implementation is concise, its efficiency can be further improved.

There are two sources of the inefficiency. The first is the difficulty of providing appropriate time boundary conditions for the Datalog query. This is necessary in real implementation because without a reasonable boundary condition, the recursive process may not terminate.

The second issue is the redundant calculation introduced by the naive implementation. Although the user only needs to know the area with property P in one or several separate time instances, the naive implementation always calculates these areas for every time unit during a time period. That is an extra burden for the system that can be avoided.

For example, a user wants to know the area that has property P at time T. If the user decide to limit the depth of the recursive execution to 10 weeks, the naive implementation will try to find all such areas at every week from week $T-10$ to week T. The results between week $T-10$ to week $T-1$ are neither necessary nor confident.

In order to improve the efficiency of the algorithm, we modify the naive Datalog solution as follows:

Theorem 3.1

$$
\begin{array}{lll}
P & = & \{(x,y,t) \ \mid \ A(x,y,t) \ \ \vee \\
 & & \ (\bigvee_{m=1}^{+\infty} (C(x,y,t,m-1) \wedge A(x,y,t-m)))\}
\end{array}
$$

where

$$
\begin{array}{lll}
C = \{(x,y,t,m) & \mid & (B(x,y,t) \wedge m = 0) \ \vee \\
 & & (B(x,y,t-m) \wedge \\
 & & \ C(x,y,t,m-1) \wedge m \geq 1)\}
\end{array}
$$

Theorem 3.1 allows us to express the problem of finding relation P by an efficient Datalog query as follows.

$$
\begin{array}{lll}
C(x,y,t,0) & :- & B(x,y,t). \\
C(x,y,t,m) & :- & B(x,y,t-m), \\
 & & C(x,y,t,m-1),\ \ m \geq 1. \\
 & & \\
P(x,y,t) & :- & A(x,y,t). \\
P(x,y,t) & :- & C(x,y,t,m-1), \\
 & & A(x,y,t-m),\ \ m \geq 1.
\end{array}
$$

Note: Relation $A(x, y, t)$ and $B(x, y, t)$ are defined in *Query 1*. Assume $C_k = \{(x, y, t) \mid C(x, y, t, k)\}$, we always have $C_j \subseteq C_i$ for all $1 \leq i < j$. That means for each fixed time t the area of $C(x, y, t, m)$ monotonously decreases as m increases.

Hence the Datalog query evaluation of the P should terminate after some finite number of rule applications. The users can easily assign an appropriate constant M as the upper bound of m for their specific application to balance the accuracy and calculation time. The bigger M, the more accurate the result and the more the calculation.

After this improvement, the execution time of the optimized Datalog is more predictable and much less than the naive implementation in general. Detail comparison is reported in Section 4.3.

4. Sample applications of the IVSRDC

The IVSRDC system provides a general method to visualize recursively defined concepts. It can be easily applied in many different research areas. In this section, we describe two sample applications which are implemented by the IVSRDC system. Then we compare the performance of naive and optimized algorithm based on the drought analysis problem.

4.1. Drought analysis problem

Although the concept of drought is well-known for most people as a deficit of precipitation, it is hard to precisely define the beginning and ending time of a drought event. Precipitation has to be combined with time and location to represent a drought condition. Meteorologists have developed many drought indices to help the analysis of drought. Standardized Precipitation Index (SPI) [5] is one of the common and simple measures of drought. The original SPI data is calculated by the entire precipitation data stored in a point-based spatio-temporal database sampled in weather stations, that is, only the sample points have SPI values.

We use SPI values to calculate the drought regions. Values of SPI range from 2 and above (extremely wet) to -2 and less (extremely dry) with near normal conditions ranging from 0.99 to -0.99. McKee et al. [5] defined the criteria for a *drought event* for any time scale as follows. A drought event occurs “any time the SPI is continuously negative and reaches an intensity of -1 or less. The event ends when the SPI becomes positive.” Therefore, each drought event has a duration defined by its beginning and end, and an intensity for each time unit that the event holds.

We describe the problem in the format of Definition 1.2 as follows:

Definition 4.1 An area is in $drought$ during week T if during T we measure either

(i) -1 or less SPI value or
(ii) between -1 and 0 of SPI value ***and*** it was in $drought$ during week $T - 1$.

In Section 2, we provide a general solution for this kind of problems. By inputing the three arguments as $k = -1, k_1 = 0$ and $T =$ request time, the system can automatically generate a correct Datalog query to find the drought areas and visualize the result in a picture.

Figure 7 on the next page shows the evaluation process of this recursive Datalog query. The red area represents the A relation. Green area is the B relation and blue area is the area of C relation.[1] For any integer $m_i < m_j$ we have $C_{m_i} \subseteq C_{m_j}$.

4.2. Air pollution evaluation problem

Clean air is considered to be a basic requirement for human health and well-being. The development of toxicity is a complex function of the interaction between a pollutant concentration and the exposure duration. After peak exposure for a short period, a pollutant may cause acute, damaging effects. Exposing to a lower concentration of a pollutant for a long period of time may cause irreversible chronic effects. It is easier to evaluate the effects of a short-term peak exposure to a chemical than a prolonged exposure to a lower concentration. However, in some cases, the low concentration over a long period can have more impact than the pattern of peak exposure [10].

As stated in [10], a similar situation occurs for effects on vegetation. Plants are generally damaged by short-term exposures to high concentration as well as by long- term exposures to low concentration. Therefore, both short-and long-term guidelines to protect plants are proposed. In this paper, we focus our work on the effects of polluted air on plants because the exposure - response relationship between pollutants and plants is more accurate than that on human health. However, the rules can be easily applied on human health if required.

We can evaluate and visualize the air pollution based on the safe and critical level of the pollutants given in the air quality guidelines [10].

When the pollutant concentration exceeds a critical level, the probability of damage is considered to be non-zero. That implies a non-sustainable force. This force can lead to actual damage at any point in time.

If the concentration keeps below the safe level, then the pollution is safe for most acceptors. No adverse effects were found in the receivers under this condition.

When the concentration level of the pollutant is between the safe level and the critical level, then there is no immedi-

1 This figure looks best if printed in color. In a black-and-white printout, the blue areas will be black and the red areas gray.

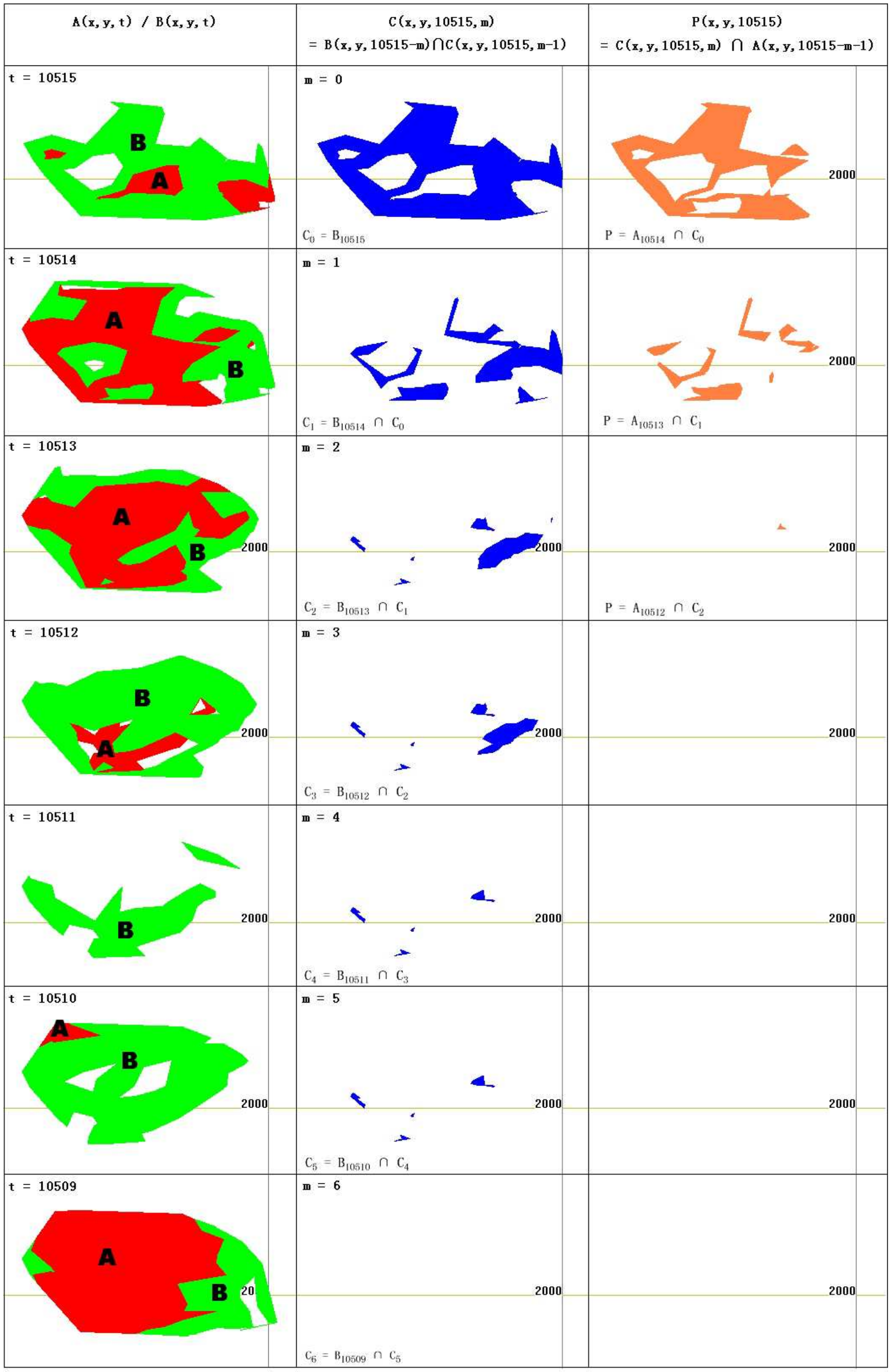

Figure 7. Process of finding the drought places at time 10515 (the 27th week of year 2001).

ate damage caused by the pollutant. But the plant may accumulate this chemical in the body and produce adverse effects after a long period of time.

It is also important to know that the critical and safe levels may not be unique in different regions. For example, the critical levels of sulfur dioxide (SO_2) in forest or vegetation areas are almost half of that in the agricultural crop areas. [10] provides a table of the critical levels for the effects of sulfur dioxide on vegetation, which is listed in Figure 8. To handle this problem, we can apply the general solution for each region and combine the outputs together in the final output.

Vegetation category	Guideline (μg/m^3)	Time period	Constraints
Agricultural crops	30	Annual and winter mean	
Forests and natural vegetation	20	Annual and winter mean	
Forests and natural vegetation	20	Annual and winter mean	(a)
Lichens	10	Annual mean	
Forests	1.0	Annual mean	(b)

(a) Accumulated temperature sum above +5°C is $< 1000°C \cdot$ *days per year.*

(b) Where ground level cloud is present $\geq 10\%$ *of time*

Figure 8. Guidelines for the effects of sulfur dioxide (SO_2) on vegetation: critical levels [10].

In this paper, we choose sulfur dioxide (SO_2) as the sample pollutant to visualize the air pollution in IVSRDC system. The area we evaluate here has four regions. The green region on the left represents natural vegetation area. The golden area on the right-bottom of the map represents agricultural crops. The dark green top-right area is a forest. The blue region in the map is a river. The red point in the middle of the map represents a factory which releases SO_2 into the air every day. Each of these object is represented by a spatial constraint relation in the input database and specified by the user through the web page interfaces.

The long-term pollution area can be defined as follows:

Definition 4.2 An area is $polluted$ during week T if during T we measure either
(i) $critical$ level or more pollutant in the air or
(ii) between $safe$ and $critical$ level of pollutant in the air *and* it was $polluted$ during week $T-1$.

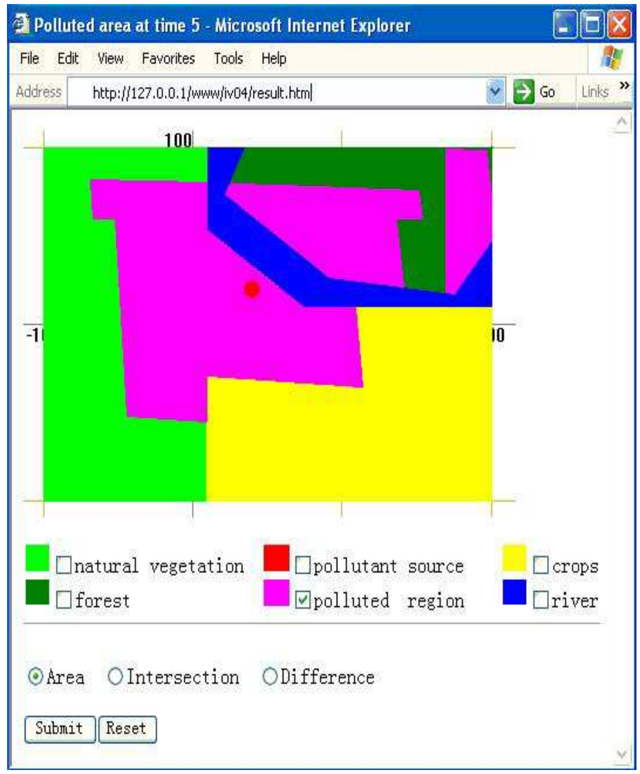

Figure 9. Output web page of finding polluted regions.

Applying the general solution for each region and combine them together, we can get the output map of the polluted regions. Figure 9 is the final output web page with the output image combined with the polluted regions, different plant regions, river object, and factory object. The user can calculate the area of each object by selecting it from the legend. And he/she can even calculate the intersection or difference of any two objects.

4.3. Performance analysis

To evaluate the improvement of the optimization, we do experiment on the drought analysis problem and show the execution result in Figure 10.

The constraint database in our experiment is the interpolated SPI data of year 2001. We use SPI data in the evaluation because it is come from real data and is representative for other problems. It has 4264 tuples, each of them has three inequality functions and one equality function. As a preparation of the data, we generate relation A and B for both algorithms first. Relation A has 608 tuples and relation B has 2511 tuples.

Bound	Items	Naive Implement	Optimized Implement	
			C(x,y,t,m)	P(x,y,t)
10489-10499	time	1060sec	669sec	48sec
	tuples	80	150	0
10499-10509	time	1020sec	324sec	28sec
	tuples	217	86	59
10505-10515	time	$\geq$ 3h	2153sec	118sec
	tuples	N/A	218	99
10519-10529	time	4706sec	325sec	18sec
	tuples	397	40	0
10529-10539	time	2791sec	1183sec	67sec
	tuples	97	203	5

Figure 10. Efficiency comparison in finding drought regions.

In practice, the naive implementation is inefficient and the execution time is unpredictable. In some cases, it generates no output before it uses out the memory of the testing machine. On the other hand, the optimized implementation can always generate an output in an acceptable time. Its execution time is much more predictable than the naive algorithm.

5. Conclusion and future works

For spatio-temporal applications, recursive queries are not expressible using the basic query languages of GIS systems. Some relational database and knowledge-based systems provide recursive queries, but they do not provide spatio-temporal data representation. Hence the visualization of recursively defined concepts cannot be handled by any known system in an easy way. They would usually require some special functions to be written in a programming language like C or C++ and added to a library. In contrast, the IVSRDC only uses standard SQL and Datalog queries to solve the problem. Therefore, the program is simple, declarative and high-level query that is easy to maintain.

This feature is important, because the requirement of visualizing recursively defined concepts on spatio-temporal data is frequent enough to need a general and simple solution method as shown in Example 5.1.

Example 5.1 An ecosystem is in *danger during month* T if during month T the density of one important plant species either
(i) decreased 10% or more or
(ii) decreased between 2% and 10% and it was already in danger during month $T-1$.

Constraint databases integrate database technology with constraint solving methods to visualize complex spatio-temporal problems [8]. Based on the recursive Datalog query language and the MLPQ system, we can visualize some novel queries in a simple and efficient way, which would be difficult or impossible to do with other systems. The problems solved by those queries are not trivial and can be found in many other important research areas.

We are currently extending the usage of the IVSRDC system. For example, we are improving the user interfaces and allow the user to specify the color of each overlay object. We are expanding the *recursive Datalog Generator* so that the user can select an object and compare the difference of its areas at different time instances. Besides the improvement of the IVSRDC system, we are also branching out to other new applications that require displaying recursively defined spatio-temporal concepts.

References

[1] J. E. Goodman and J. O'Rourke. *Handbook of Discrete and Computational Geometry*. CRC Press, 1997.

[2] L. Li, Y. Li, and R. Piltner. A new shape function based spatiotemporal interpolation method. In *1st International Symposium on the Applications of Constraint Databases*, number 3074 in Lecture Notes in Computer Science, pages 25–40. Springer, 2004.

[3] L. Li and P. Revesz. Interpolation methods for constraint-based spatio-temporal geographic data modeling. *Journal of Computers, Environment and Urban Systems*, 28(3):201–227, 2004.

[4] P. A. Longley, M. F. Goodchild, D. J. Maguire, and D. W. Rhind. *Geographic Information Systems and Science*. Wiley, 2001.

[5] T. B. McKee, N. J. Doesken, and J. Kleist. The relationship of drought frequency and duration to time scales. In *8th Conference on Applied Climatology*, pages 179–184, 1993.

[6] P. Revesz. *Introduction to Constraint Databases*. Springer, 2002.

[7] P. Revesz and L. Li. Constraint-based visualization of spatial interpolation data. In *6th International Conference on Information Visualization*, pages 563–569. IEEE Press, 2002.

[8] P. Revesz and L. Li. Constraint-based visualization of spatiotemporal databases. In M. Sarfraz, editor, *Advances in Geometric Modeling*, pages 263–276. John Wiley Inc., 2003.

[9] J. R. Shewchuk. Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator. In *Workshop on Applied Computational Geometry*, number 1148 in Lecture Notes in Computer Science. Springer, 1996.

[10] WHO-Europe. *Air quality guidelines for Europe, Second Edition*. Number 91 in European Series. World Health Organization Regional Publications, 2000.