A Theory of the Spatial Computational Domain

Shaowen Wang [1] and Marc P. Armstrong [2]

[1] Academic Technologies – Research Services and Department of Geography, The University of Iowa
Iowa City, IA 52242
Tel: +1 319-335-6713
FAX: +1 319-335-5505
Email: shaowen-wang@uiowa.edu

[2] Department of Geography and Program in Applied Mathematical and Computational Sciences, The University of Iowa
Iowa City, IA 52242
Tel: +1 319-335-0153
FAX: +1 319-335-2725
Email: marc-armstrong@uiowa.edu

Abstract

Most parallel processing methods developed for geographic analyses bind the design of domain decomposition and task scheduling to specific parallel computer architectures. These methods are not adaptable to emerging distributed computing environments that are based on Grid computing and peer-to-peer technologies. This paper presents a theory to support the development of adaptable parallel processing methods for geographic analyses performed on heterogeneous parallel processing environments. This theory of the spatial computational domain represents the computational intensity of geographic data and analysis methods, and transforms it into a common framework based on transformation theories from earlier cartographic research. The application of the theory is illustrated using an inverse distance weighted interpolation method. We describe the underpinnings of this analysis, and then address the latent parallelism of a conventional sequential algorithm based on spatial computational domain theory. Through the application of this theory, the development of domain decomposition methods is decoupled from specific high performance computer architectures and task scheduling implementations, which makes the design of generic parallel processing geographic analysis solutions feasible.

1. Existing Problems in Parallel Processing of Geographic Information

The state-of-the-practice in parallel processing of geographic information binds the design of domain decomposition and task scheduling to specific conventional parallel computer architectures. This tight-coupling approach is problematic to software design for three reasons:

1.) Domain decomposition and task scheduling methods focus on the characteristics of spatial data and operations performed on them.

2.) Any change to spatial data or operations requires a corresponding change in the design of domain decomposition and task scheduling methods.
3.) Domain decomposition and task scheduling strategies depend upon specific parallel architectures.

These three problems are analogous to those observed in graphics programming before the advent of device-independent software. In this case, these problems hinder the development of portable parallel geographic analysis methods based on computational Grids (see endnote). To eliminate these problems, we introduce a new spatial computational domain theory and illustrate an application using the TeraGrid (TeraGrid, 2005).

2. Spatial Computational Domain Theory

The spatial computational domain is formally defined to comprise several two-dimensional computational intensity surfaces. Given a spatial domain projected to a two-dimensional (x and y) Euclidian space, each two-dimensional computational surface can be represented as a vector $\mathbf{c} = (c_{ij})$ in the space $\mathbf{R}^N$, where $N = x_c \times y_c$, and where x_c is the number of cells in the x dimension, and y_c is the number of cells in the y dimension of the computational surfaces. Each component of $\mathbf{c}$ corresponds to the computational intensity at cell (i, j).

The spatial computational domain is similar to an image obtained from an evaluation of the following function:

$$f: I^2 = [0, 1] \times [0, 1] \; ? \quad \mathbf{R} \tag{1}$$

at cells $(i/x_c, j/y_c) \in I^2$, $i = 1, \ldots, x_c$, $j = 1, \ldots, y_c$. Thus,

$$c_{ij} = f\,(i/x_c, j/y_c) \tag{2}$$

The value of c_{ij} represents the computational burden derived from a new computational transformation concept that is consistent with the role of transformations in other domains of GIScience (Tobler, 1979):

- Computing time: the time taken to perform the analysis within cell (i, j);
- Memory: the memory required to perform the analysis within cell (i, j); and
- I/O: data input/output and transfers to perform analyses within cell (i, j).

3. Computational Transformation

A computational transformation elucidates the computational intensity of a particular geographic analysis based on the characteristics of spatial data and operations. Two types of computational transformations are identified:

1.) Data-centric functions transform spatial data characteristics into memory or I/O surfaces.

2.) Operation-centric functions consider the spatial operations that are directly or indirectly related to spatial data characteristics, and transform the characteristics of spatial operations into a computing time surface.

3.1 An Illustrative Example

An operation-centric transformation function illustrates the spatial computational domain for

Clarke's inverse distance weighted (IDW) interpolation algorithm (Clarke, 1995). This domain consists of a single computing time surface because both memory and I/O requirements are trivial even if a common desktop PC (e.g., 1G RAM and 2GHz Pentium processor) is used to execute the algorithm. The transformation function - *IDW-of* on the cell (*i, j*) is defined as follows:

$$IDW\text{-}of_{(i,j)} \; ? \; timeUnit \times \frac{ne_{(i,j)}}{(ns_{(i,j)}+1)\times\sqrt{dp_{(i,j)}+densityThreshold}} \tag{3}$$

where *ne* represents the number of empty grids that invoke a search for their *k*-nearest neighbors; *ns* represents the number of sampling points; *dp* represents the local density of sampling points; *timeUnit* is a constant used to convert the value to a computing time unit (e.g., seconds); *densityThreshold* is a constant that represents the minimum point density in any given region of the domain considered, and prevents the denominator from becoming zero. The value of $dp_{(i,j)}$ is calculated using the following equation:

$$dp_{(i,j)} = \frac{ns_{(i,j)} + \sum_{k=1}^{number_{neighbor}} ns_k}{ns_{(i,j)} + ne_{(i,j)} + \sum_{k=1}^{number_{neighbor}} (ns_k + ne_k)} \tag{4}$$

where ns_k represents the number of existing sampling points within the (*i, j*) neighbor cell indexed as *k*; and $number_{neighbor}$ represents the number of (*i, j*) neighbors being considered.

Figure 1 shows a synthetic dataset with 2000 sampling points. The raster structure (2000 by 2000) is created in the initialization step of Clarke's algorithm. Figure 2 shows a 3-D plot of

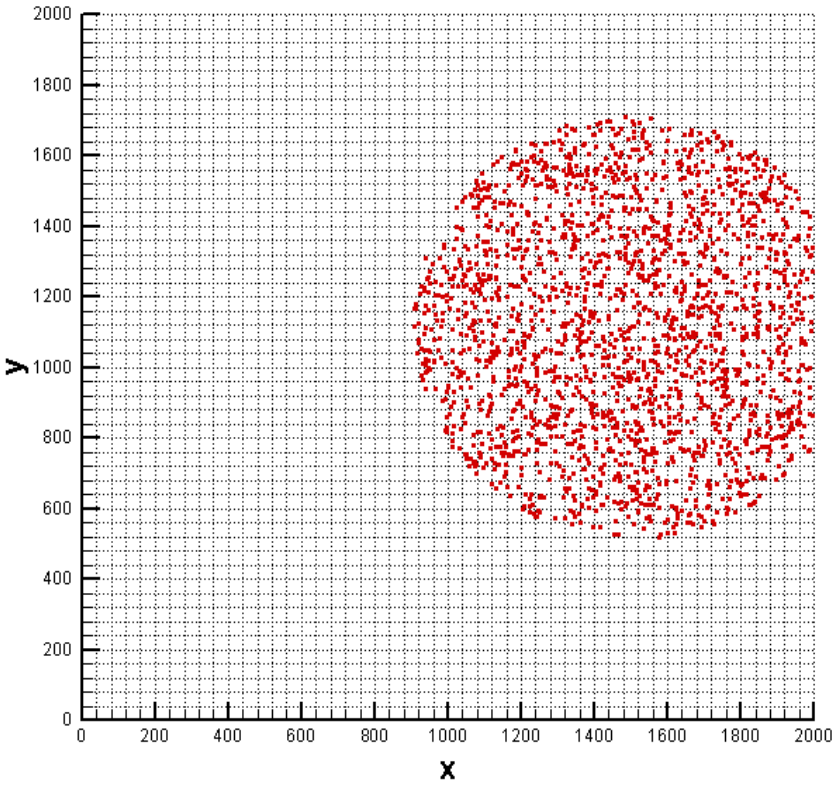

Figure 1. A dataset of two thousand points, one cluster with a uniform random distribution

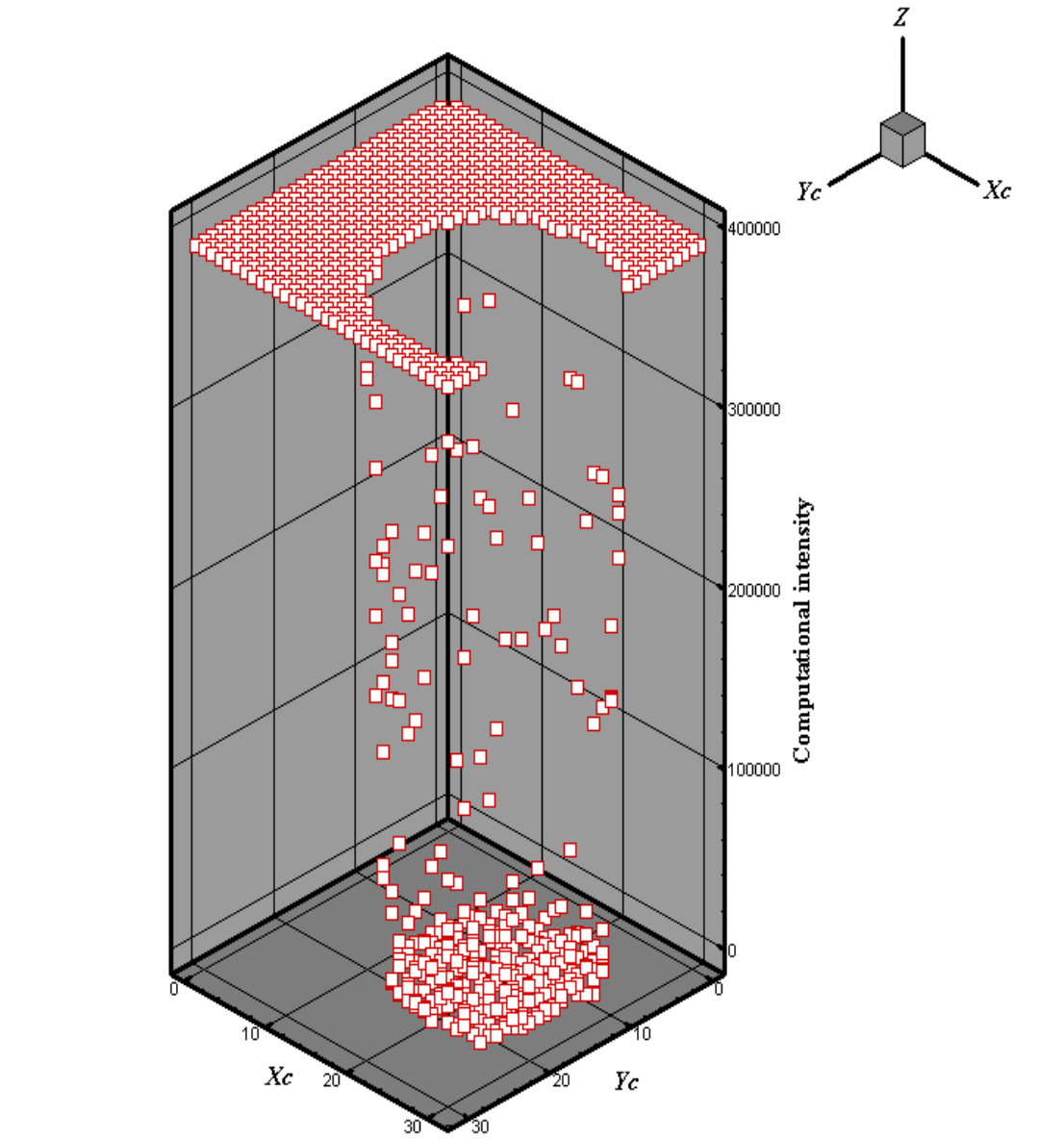

Figure 2. A 3-D plot of the spatial computational domain for Clarke's algorithm using the dataset in Figure 1; cell size is specified as 62.5 × 62.5, and thus the compution tal domain dimension is fixed at 32 × 32 given a 2000 × 2000 data domain dimension.

the spatial computational domain derived from this dataset using the computational transformation function (Equation 3). Based on a decompositio n of the spatial computational domain using a quadtree method (Wang and Armstrong, 2003), a parallel interpolation of this dataset using the TeraGrid (Table 1) has a speedup of more than 20 times compared to a sequential algorithm executed on the fastest single processor of the TeraGrid. Table 2 shows the computing time taken by the parallel interpolation analysis in a set of computational experiments in which the spatial computational domain was decomposed to create different numbers of parallel jobs. These jobs are scheduled to available TeraGrid resources to achieve load balancing based on computational

intensity information derived from their spatial computational domain. This is a preliminary result; we are devising additional experiments to evaluate performance.

Table 1. Hardware information about TeraGrid resources used

Sites	Compute nodes	CPU	Intra-network	Disk	OS	Local resource manager
ANL/ UC	Dual-processor: 96 nodes; RAM memory: 4GB	Intel® Xeon® 2.4GHz	Myrinet 2000, Gigabit Ethernet, Fiber Channel	4TB Network File System	Linux 2.4.21-SMP (shared memory multi-processor)	PBS
	Dual-processor: 16 nodes; RAM memory: 4GB	Intel® Itanium® 2, 1.3 GHz	Same	4TB Network File System	Linux 2.4.21-SMP	Same
Caltech	Dual-processor: 52 nodes; RAM memory: 6GB	Same	Same	75 TB Parallel Virtual File System	Linux 2.4.19 - SMP	Same
NCSA	Dual-processor: 128 nodes; RAM memory: 12GB	Same	Same	5TB Network File System	Linux 2.4.21-SMP	Same
	Dual-processor: 128 nodes; RAM memory: 4 GB					
SDSC	Dual-processor: 128 nodes; RAM memory: 4GB	Same	Same	1.6TB Network File System	Linux 2.4.19 - SMP	Same

4. Concluding Discussion

Our spatial computational domain theory has solved several problems present in previous GIScience parallel processing research because it provides computational intensity representations that are required for efficient parallel processing of geographic analyses. At the same time, this theory has paved the way to establish generality in domain decomposition methods. The development of these functions is independent of any computer architecture, which also indicates the theory can be used to address problems with architectural incompatibilities. Our ongoing research focuses on the application of this theory to develop geo-middleware that supports other types of Grid-based geographic information analyses.

Table 2. Interpolation time in the TeraGrid environment

TeraGrid sites	Number of computing jobs submitted to each TeraGrid site			
ANL/UC	230	57	260	68
NCSA	138	44	219	71
SDSC	113	34	161	74
Caltech	62	25	106	38
ANL/UC	1	0	2	5
Computing Time (hh:mm:ss)	**00:39:46**	**00:44:19**	**00:42:21**	**00:48:32**

Note

Grid computing has been the recent focus of a substantial amount of research and development activity. Though its basic concepts were stated earlier, the term Grid was coined in the late 90s by Foster and Kesselman (1999) to describe a set of resources distributed over wide-area networks that can support large-scale distributed applications. The analogy likens the Grid to the electrical power grid: access to computation and data should be as easy, pervasive, and as standardized as plugging an appliance into an outlet (Foster and Kesselman, 1999). In the foundational paper "The Anatomy of the Grid", Foster *et al.* (2001) more formally define the Grid as a set of environments for coordinated resource sharing and problem solving in dynamic, multi-institutional, virtual organizations. This coordination is orchestrated using protocols and specialized software referred to as Grid middleware. In that paper, they go on to argue that the Grid is central not only to "e-science", but also to industry and commerce, where coordination of distributed resources both within and across organizations has become an important business activity.

5. References

Clarke, K. C., 1995, *Analytical and Computer Cartography*, *Second Edition* (Englewood Cliffs, NJ: Prentice-Hall).

Foster, I., Kesselman, C., and Tuecke, S. (2001) The anatomy of the Grid: enabling scalable virtual organizations. *International Journal Supercomputer Applications*, **15**(3), available at: http://www.globus.org/research/papers.html.

Foster, I., and Kesselman C., 1999, *The Grid: Blue Print for a New Computing Infrastructure* (San Francisco, CA: Morgan Kaufmann Publishers, Inc.).

TeraGrid, 2005, http://www.teragrid.org/.

Tobler, W. R., 1979, A transformational view of cartography. *American Cartographer*, **6**(2), 101-106.

Wang, S., and Armstrong, M. P., 2003, A quadtree approach to domain decomposition for spatial interpolation in Grid computing environments. *Parallel Computing*, **29**(10), 1481-1504.