# A test bench for evaluating spatial indexation methods for connecting points to large networks

Anders Dahlgren[1], Lars Harrie[2]
[1] The National Rural Development Agency Östersund, Sweden,
anders.dahlgren@glesbygdsverket.se
[2] GIS Centre, Lund University, Lund, Sweden, lars.harrie@nateko.lu.se

## SUMMARY
This study deals with performance issues for connecting points to a network in geographic accessibility studies. The problem is analysed by the development of a test bench where the spatial indexation methods kD-tree, Morton code and fixed grid are implemented. Using the test bench scenarios of networks, point datasets, spatial indexation methods, and methods for computing distances between point and line segments are evaluated. A case study indicates that the indexation methods kD-tree and fixed grid have the best performance for connecting point to the network. Furthermore, the results show that an "algebraic method" is much faster than a "trigonometric method" for computing the shortest distance between the point and the network.

**KEYWORDS:** GIS, geographic accessibility, spatial index, fixed grid, kD-tree, Morton code

## 1. INTRODUCTION
Geographic accessibility studies are of interest in application areas such as: health geographics (Guagliardo, 2004), transport modelling (Berglund, 2001) and logistics (Bergqvist and Tornberg, 2005). To compute the geographic accessibility three datasets are required: the start points (e.g. census data), the target points (e.g. service location data), and a network (most commonly a road network). In principal, accessibility is computed as follows. Firstly, the starting points and the target points have to be connected to the network. Secondly, the shortest distance, or shortest travelling time, from the starting point to the target point in the network is computed.

The Swedish National Rural Development Agency (SNRDA, 2006) is responsible for monitoring the Swedish population's accessibility to different kinds of service. Examples of services of interest are: airports, hospitals and grocery stores. The SNDRA has developed a tool for these accessibility studies (Figure 1 gives an example of the output). In recent years, the datasets used as input data has become more detailed and therefore larger. Dahlgren (2005) identified that the two main processes were too slow in SNRDA:s accessibility tool: (1) connecting points to the road network, and (2) computing the shortest route in the network. This paper is concerned with improving the first the first of these processes.

The aim of this study is to evaluate methods for connecting points to networks; and, especially, to evaluate spatial indexation methods. To study the spatial indexation methods in an isolated environment a test bench is established. All computations in this study are performed on this test bench. Since the programming environment is the same in this text bench as for the accessibility tool, it will be straight forward to utilize the code also in the production tool.

This paper is organised as follows. In section 2, the methodology is given for connecting points to the network. Especially, spatial indexes and formulas for computing the distance between point and lines are described. Section 3 describes a test bench where these indexation methods and distance formulas are implemented. An evaluation of the implemented methods is given in Section 4. The paper concludes with a discussion and conclusions.

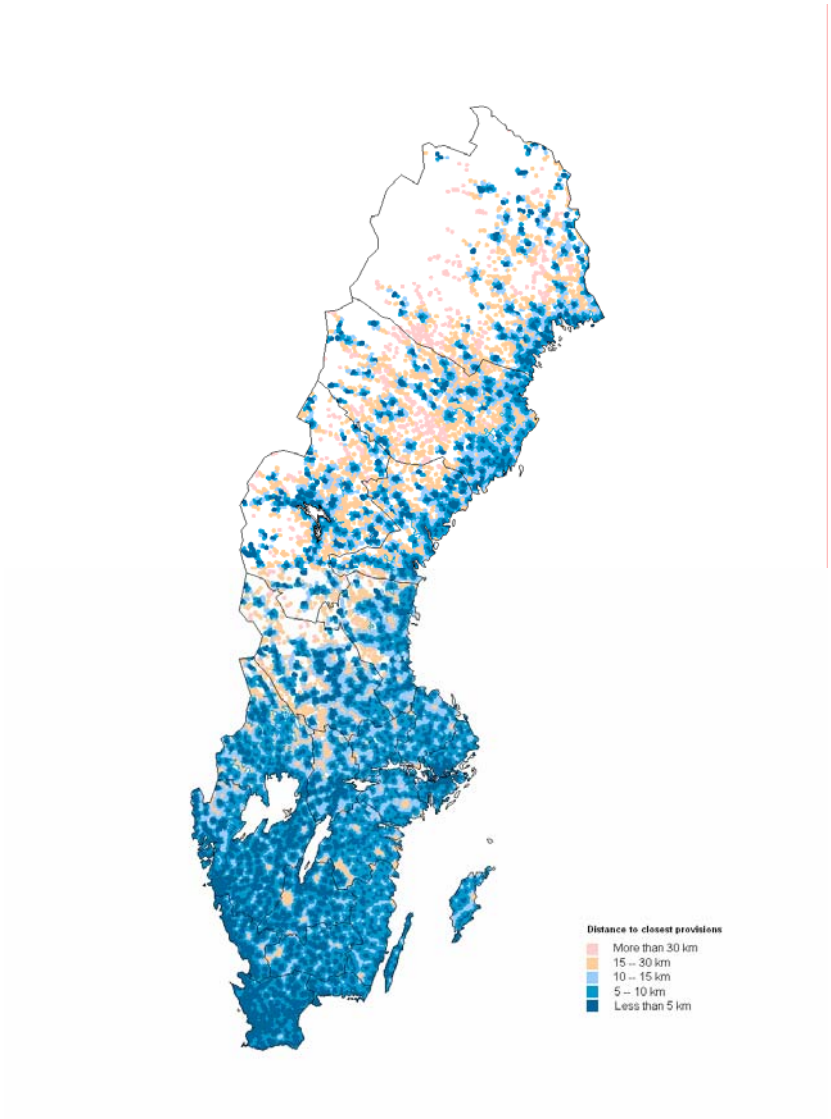



***Figure 1:*** A thematic map showing an example of the output NRDA proximity application showing the Swedish populations (2004) accessibility to the closest grocery store. Copied from SNDRA (2005).

**2. METHODS FOR CONNECTING POINTS TO THE NETWORK**

If we do not have to consider performance issues it is easy to connect a point to a network. It is only to compute the distance between the point and all line segments in the network, and connect the point to the closest line segment (Figure 2). However, in many applications the number of line segments is abundant which causes the connection process to be slow. In this section two issues for improving the efficiency are described: *spatial indexation methods* and *formulas for computing the distance between the point and the line segment.*

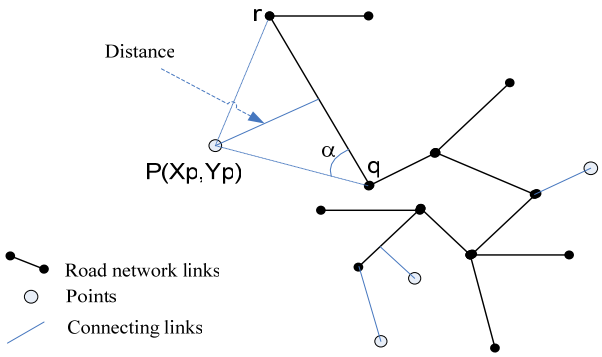



***Figure 2:*** Connecting points to a network. The points are connected to its closest point on the network via the connecting links.

## 2.1 Spatial indexation methods

To use indexes in the Cartesian plane is difficult since the plane lacks ordering. There are a number of methods to circumvent this problem, e.g. by introducing an artificial ordering (by using a space filling curve) or introduce tree structures (kD-trees, R-trees, etc.) (see overviews in e.g. Rigaux et al., 2002 or Worboys and Duckham, 2004). The indexation methods can be broadly categorized as either data-driven or space-driven. In space-driven methods each location in space is indexed irrespective of the data distribution. A number of researchers have worked with optimising the size of the indexation cells (e.g. Ottoson and Hauska, 2002). In data-driven methods the indexes is dependent on data distribution. This is often advantageous if the data is unevenly distributed.

By using an indexation method, the search process is divided into two parts. In the first part, a search is performed (by using the index). The result of this part is a candidate set (in our application this candidate set consist of a number of links). In the second part a linear search is performed in the candidate set to identify the best element (in our application the closest link to the point). To be successful an indexation method should create a small candidate set rapidly. Some indexation methods, e.g. the space filling curves, do not guarantee that the best element is part of the candidate set. This implies that for these indexation methods there is a trade off between fast computation (i.e. creating small candidate sets) and the likelihood of getting the correct answer.

Even though there is an extensive literature of indexation methods, there are few studies published that concerns design and implementation issues (Hadjieleftheriou et al., 2005). The aim of our study is to implement some well-known indexation methods and evaluate their performance for connecting points to a network. The following methods are implemented:

- no index (only used for reference),
- fixed grid ordering (space-driven method),
- kD-tree (data-driven method), and
- Morton code (a space filling curve; in our implementation we use it as a space-driven method).

A short justification of the choice of methods is as follows. Firstly, the *kD-tree* is a chosen because it is interesting to see if it is the kD-tree implementation in the current accessibility application that causes the bad performance or if it is the indexation method itself. Secondly, *Fixed grid ordering* is a conceptually easy method that should give good results in our application because of the relatively even spread spatial datasets used in the accessibility analysis. Finally, we wanted to evaluate a space filling curve method and the choice became *Morton code*.

*SEARCHES WITH NO INDEX*

As a reference, a function for connecting the points in the point dataset to the road network without indexation is implemented. This means that the shortest distance from each point in the point dataset is calculated to every link in the road network.

*Fixed grid ordering*

One method for indexing spatial data is to order it in a fixed grid (Figure 3). Each spatial object is sorted into a "bucket" if it is located within the corresponding cell. A special case that must be handled is when a link in the road network crosses two or more cells. In the test bench, this special case is treated by dividing long links and adjusting the grids to a fitting size. When a point is to be connected to the road network, the search begins in the cell where the point is stored and in the adjacent cells. If no link is found in these cells, the search expands its outer perimeter with one cell in a new iteration, if no link is found in that iteration the whole network is searched.

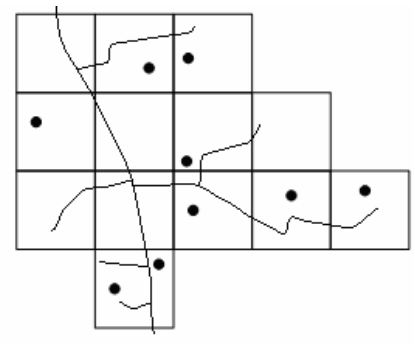


***Figure 3:*** A map that shows points and links overlaid by fixed grid.

*KD-TREE*

The kD-tree indexation method uses a tree structure to index the spatial point objects (Bently, 1975; Figure 4). The kD-tree is a variant of a binary search tree that divides the data on the *x*-coordinate on uneven levels of the tree and the *y*-coordinate on the even levels. Working its way down to the leaves where the coordinates of the spatial objects are stored. In the test bench implementation, the start points of the links are stored in leafs of the tree. In the search for candidates the longest link in the network is used to define the search window.

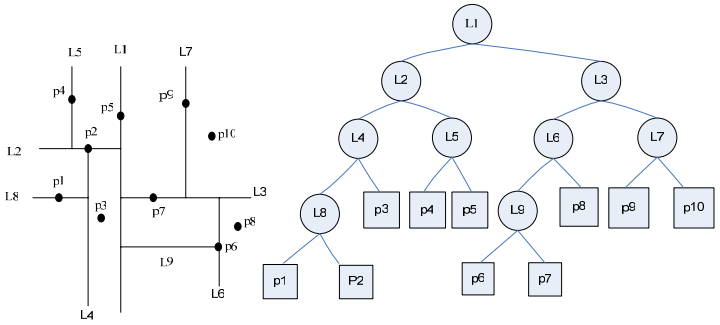



***Figure 4:*** A kD-tree: in the left figure the plane is subdivided, and in the right the corresponding binary tree is illustrated. (The figure is redrawn from de Berg et al., 2000, p. 100).

*MORTON CODE*
The Morton Code or Z-ordering is a type of space filling curve; the space-filling curves are indexation method that partially preserves the proximity relations between spatial objects (cf. Chen and Chang, 2005). The idea is to construct a curve that runs through the plane with the ideal property that areas that are close to each other in the plane also are close to each other along the curve (Figure 5). Morton code can be computed by bit interleaving of binary representations of the coordinates (see e.g. Shekhar and Chawla, 2003).

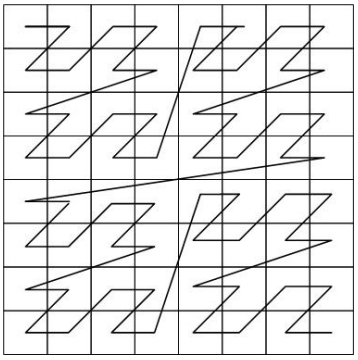


***Figure 5:*** An illustration of the Morton code**.**

## 2.2 Computing the shortest distance between a point and a line segment

A point is, in our application, connected to a network via the closest point on the network (Figure 2). The shortest distance between a link and a point is either the distance to a node/breakpoint or an interior point of a line segment. The distance to the node/breakpoint is easily computed by Pythagorean theorem. The distance (*dist*) between point *p* and the interior of a line segment (with the end points *q* and *r*) can be computed by a trigonometric method (using the notations in Figure 2):

$$\cos\alpha = \frac{d(q,p)^2 + d(q,r)^2 - d(r,p)^2}{2 * d(q,p) * d(q,r)}$$
$$dist = d(q,p) * \sin\alpha \qquad (1)$$

where $d(.,.)$ is the Euclidean metric. Since trigonometric functions are slow in computer calculations, we also implemented the following algebraic methods (see e.g. Worboys and Duckham, 2004; Okabe and Miller, 1996) in the test bench:

$$dist = \frac{\left|(a \cdot x_p + b \cdot y_p + c)\right|}{\sqrt{a^2 + b^2}} \qquad (2)$$

where $x_p$ and $y_p$ are the coordinates for the point *p*,
|…| denotes absolute value, and
the line segment is represented as part of the infinite long line (*l*):

$$l : \left\{(x,y) \;\middle|\; a \cdot x + b \cdot y + c = 0\right\}. \qquad (3)$$

To avoid the square root in the denominator in Equation 2, we always use the square of the distance in the computations.

## 3. A TEST BENCH FOR SPATIAL INDEXATION METHODS

A test bench, called Geographic Index Laboratory (GIL), was developed to evaluate scenarios of input datasets and spatial indexation methods in detail and isolated from interference from the main tool. The two algorithms for computing the distance between a point and a line segment (Equations 1 and 2) as well as the four indexation methods were implemented. GIL is an open source application developed in C# and its source code can be downloaded from Dahlgren (2006). The application can be used as general education tool teaching spatial indexation methods.

The user interface of the test bench is shown in Figure 6. The controls of the different indexation methods are placed in the upper left corner. Information of progress and performance is given in the lower left corner. The panel at the right shows a map over the datasets.

The test bench is run as follows. First, the user opens two datasets, a network dataset and a point dataset. In the first calculation step, GIL creates a spatial index for the line segmants in the network. In a second step, the indexation methods is used to connect the points in the point dataset to the network. The result (the connecting links between points and the road network links, cf. Figure 2) can be viewed in the map window or exported and analysed outside the test bench in a standard GIS environment. Information about each operation (e.g. calculation times, number of elements in the candidate sets, etc.) can be read in the lower left panel (see Figure 6).

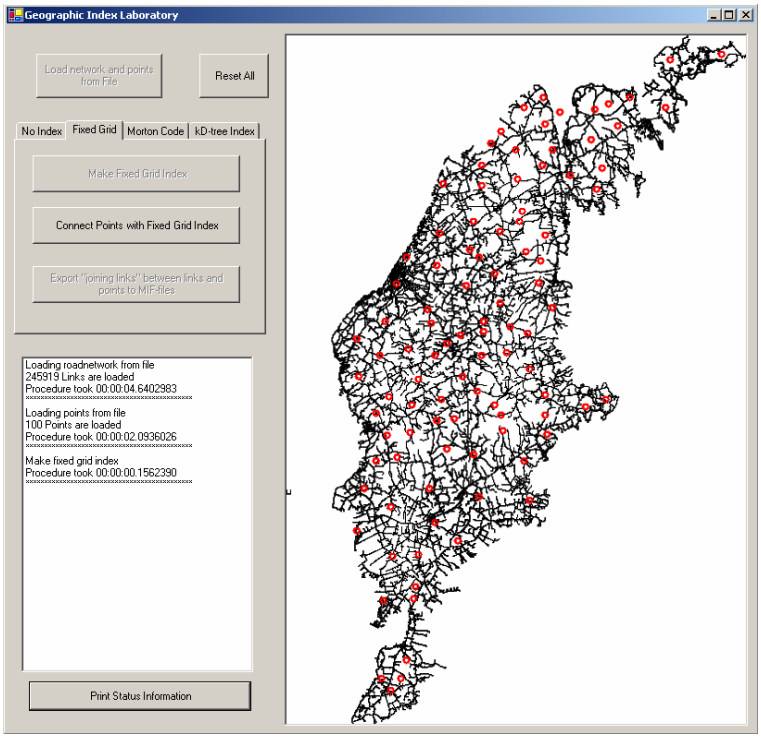



***Figure 6:*** The user interface of the Geographic Index Laboratory.

## 4. EVALUATION

Three cases studies were conducted. The first concerns spatial indexation methods, and the second methods to compute distances between point and line segments. The third case study aims at studying if the use of spatial indexation methods introduces any error in the computations. All the case studies where performed using the Geographic Index Laboratory on a PC with Windows XP, Pentium 4, 3 GHz and 2 GB Ram.

### 4.1 Comparison of spatial indexation methods

In the first case study we performed two tests. In the first test, we used the following data sets:
1) a road network from the Swedish National Road Administration (SNRA) covering the south and middle of Sweden (3 993 284 links), and
2) a regular grid point dataset (population) with a resolution of 1 kilometre covering the road network (22006 points).

All the points were connected to the network with the result given in Table 1. The calculations begin with a pre-process where the indexes are built. This process is not time critical since it only has to be run once for a network. The interesting results are given in third and forth column. In the third column the number of links selected by the binary search is given. And in the fourth column the total time for search is given. By exporting the links that connect the point with the road network the errors that accurse can be studied. All spatial index methods have parameters that can be tuned balancing exact result on one hand and good performance on the other. A method that gives acceptable performance and acceptable results for a dataset could give unacceptable results with another dataset. This is indicated in column 5.

***TABLE 1:*** TIME FOR CONNECTING POINTS TO THE NETWORK USING THE TEST BENCH.

| Indexation method | Time for building the index | Number of links in the candidate set | Time for connecting points | Gives acceptable results |
|---|---|---|---|---|
| No index | -------- | 87 876 207 704 | More than 2 h | (Yes) |
| Fixed grid | 2 sec | 24 977 155 | 31 sec | Yes |
| kD-tree | 1min 31 sec | 51 303 823 | 2 min 9 sec | Yes |
| Morton code | 49 sec | 108 388 022 | 2 min 9 sec | Yes |

In the second test we used the following data sets:
1) an unevenly distributed road network from SNRA covering a part of northern Sweden (1 401 721 links), and
2) a point dataset (population) with a resolution of 250 meters, of points where only the populated points are represented. This makes it an unevenly distributed, clustered dataset (12324 points).
All the points were connected to the network with the result given in Table 2.

***TABLE 2:*** TIME FOR CONNECTING POINTS TO THE NETWORK USING THE TEST BENCH.

| Indexation method | Time for building the index | Number of links in the candidate set | Time for connecting points | Gives acceptable results |
|---|---|---|---|---|
| No index | -------- | 17 274 809 604 | More than 2 h | (Yes) |
| Fixed grid | 1sec | 9 567 259 | 36 sec | Yes |
| kD-tree | 34 sec | 19 400 360 | 48 sec | Yes |
| Morton code | 20 sec | 32 046 357 | 40 sec | No |

## 4.2 COMPARISON OF METHODS FOR COMPUTING DISTANCE BETWEEN POINTS AND LINE SEGMENTS

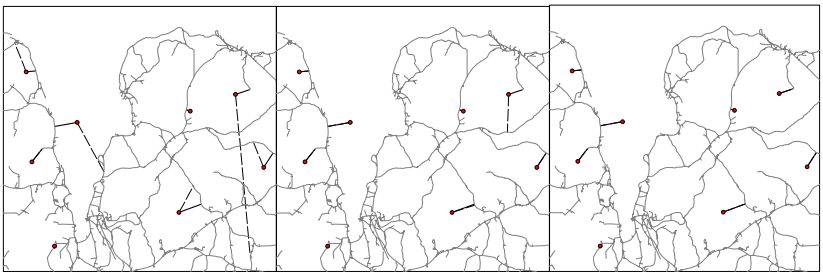
The second case study was a performance test of the two algorihms for connecting a point to the closest link. The study was performed using:

* no spatial index,
* a road network, and
* a point dataset of 100 points randomly distributed within the convex hull of the road network. This implied that 2 104 384 links were tested against all the 100 points ending up with 210 438 400 line segments to point calculations. Connecting the points with the implementation of the trigonometric function took 27 seconds (Equation 1). When using the implementation of the algebraic methods (Equation 2) the calculation took 5.6 seconds.

## 4.3 ERROR INTRODUCED BY THE SPATAIL INDEXATION METHODS

The basic idea behind all the indexation methods is that a set of candidates can be extracted using fast search methods. The question is then if we can guarantee that all the correct solution is among the candidates.

An example of the quality of the result in Morton indexes is illustrated in Figure 7. The figure shows a part of a larger road network where points are connected to the network using no indexation (illustrated by thick full lines). With this time consuming method all links in the network are tested and therefore it provides the right solution. The dotted lines show the links chosen by the Morton Code indexation method. To connect points with the Morton code indexation method a Morton Value is calculated for the point that is to be connected to the network. Then candidate links in the road network is found by looking the links that are closest to the point in Morton value order. Figure 7a shows the best links connecting the road network with the points among the average of 132 candidate links taken from a Morton value interval. As the figure indicates it gives a poor solution. When the average number of candidates is increased to 1233 (Figure 7b) a better solution is found and finally when the average number of candidates is 9600 (Figure 7c) the right solution is given.



**(a)** **(b)** **(c)**

***Figure 7:*** The connecting links given by Morton Code indexation with an increasing number of candidates.

For kD-tree and fixed grid ordering you can have the case where a point just outside search window is closer than a point inside the search window (e.g. when a point is in the corner inside the search window). Another possible problematic case is when a line segment crosses the search window, but its end points are outside the search window. These cases are quite rare and they will not imply a large error. In some empirical tests we found error ratio of linkage less than 0.1% and no linkage connection were more than 25% longer than the optimal connection.

### 5. DISCUSSION

The first case study shows that building the tree-structure for kD-tree indexation is a fairly time consuming task (Tables 1 and 2). But since the index structure can be stored between occasions when it is in use, this is of less importance. If the index must be built on each occasion the fixed grid indexation is advantegous.

The first case study also indicates that when dealing with evenly distributed datasets, using our input data, the fixed grid indexation shows a better result than the kD-tree. When the datasets are distributed unevenly the differences in performance evens out. Looking at the two index implementations the fixed grid is fairly simple to implement. It is possible to use C# collection classes (array, matrix) leaving the programmer little own code to implement. The kD-tree needs, in our programming environment, more coding and can therefore be subject for, and can gain from, further design work.

An area of interest that also should be studied further is the problem with tuning the different spatial indexation methods. The test bench allows a user to test different settings of parameters on different datasets finding the optimal settings for each occasion.

## 6. CONCLUSIONS

The aim of this study is to evaluate methods for connecting points to networks. Based on the evaluation of the case studies using the test bench we make the following recommendations:

- The recommendation for the development of accessibility tool is to implement both the *fixed grid ordering* and the *kD-tree*. The selection of the methods for a given scenario is inherent to the distribution of line segments in the network and points in the point data set. To have two indexation methods also gives the possibility for relative tests regarding performance and correct results.
- It is recommended to use the algebraic method (Equation 2) for computing the shortest distances between the points and the line segments in the network.

### ACKNOWLEDGEMENTS

We would like to thank Professor Anders Östman and Joakim Svensberg for their support and the Swedish National Rural Development Agency and the Swedish Rescue Services Agency for financing the studies.

## BIBLIOGRAPHY

de Berg, M., M. van Kreveld, M. Overmars, and O. Schwarzkopf, 2000. *Computational geometry: algorithms and applications*, 2nd edition. Springer, Berlin.

Berglund, S., 2001. *GIS in transport modelling*. Doctoral thesis. Department of Infrastructure and Planning, Royal Institute of Technology, Sweden.

Bergqvist, R., and J. Tornberg, 2005. GIS for describing and analysing regional logistics systems. *Mapping and Image Science*, Num. 2 , pp. 61-68.

Bently, J. L., 1975. Multidimensional binary search trees in database application, *IEEE Transaction on software engineering*, Vol. 5, Num. 4, pp. 333-340.

Chen, H.-L., and Y.-I. Chang, 2005. Neighbour-finding based on space-filling curves. *Information Systems*, Vol. 30, pp. 205-226.

Dahlgren, A., 2005. A proximity analysis application for large source dataset. *Proceedings of ScanGIS 2005*, Stockholm, pp. 125-136.

Dahlgren, A., 2006. *The homepage of the MapProx application.* www.mapprox.tk, 2006-02-07.

Guagliardo, M. F. 2004. Spatial accessibility of primary care: concepts, methods and challeng. *International Journal of Health Geographics*, Vol. 3, Num. 3.

Hadjieleftheriou, M., E. Hoel, and V. J. Tsotras, 2005. SaIL: A Spatial Index Library for Efficient Application Integration. *GeoInformatica*, Vol. 9, Num. 4, pp. 367-389.

Okabe, A., and H. J. Miller, 1996. Exact Computational Methods for Calculating Distances Between Objects in a Cartographic Database. *Cartography and Geographic Information Systems*, Vol. 23, No. 4, pp. 180-195.

Ottoson, P., and H. Hauska, 2002. Ellipsoidal Quadtrees for Indexing Global Geographic Data. *International Journal of Geographical Information Science*, Vol. 16, Num. 3, pp. 213-226.

Rigaux, P., M. Scholl, and A. Voisard, 2002. *Spatial databases: with applications to GIS.* Morgan Kaufmann Publishers, San Francisco.

Shekhar, S., and S., Chawla, 2003. *Spatial Databases – A Tour*. Prentice-Hall.

SNDRA, 2005. Glesbygdsverkets årsbok (in Swedish).

SNRDA, 2006. *The Swedish National Rural Development Agency*. http://www.glesbygdsverket.se/site/default.aspx, 2006-02-07.

Worboys, M. F., and M. Duckham, 2004. *GIS: A Computing Perspective*, 2nd edition. CRC Press.