Designing a proximity tool for large source datasets.

White paper

Anders Dahlgren

Swedish National Rural Development Agency, Samuel Permans Gata 2, 831 30 Östersund, Sweden

STU 03/2003-004

Anders.Dahlgren@glesbygdsverket.se

Abstract. The Swedish National Rural Development Agency has special demands on its proximity tool. This white paper describes the design decisions that where made to meet these special demands. The tool is implemented in a first version and the implementation is to be described in another white paper. The design is made on the assumption that no legacy code or third party modules are used. Meaning that an independent bottom-up design of the system can be suggested. An object-oriented view is used supporting development in an iterative process. In the end of the paper future development of the application is discussed.

The tool is in use within the Swedish National Rural Development Agency..

Introduction

The objective with this paper is to describe the tailor-made application that's been developed a for proximity analysis within the Swedish National Rural Development Agency (NRDA). The paper is addressing people with knowledge of Geographic Information Systems and has an interest in software development within that area.

This paper, together the paper Implementing a proximity tool for large source datasets (STU 03/2003-005), to be released later in 2004, are technical descriptions (white papers) of the Swedish National Rural Development Agency's tool for proximity analysis. This paper, as the name indicates, describes the main design decisions/discussions for the application.

The following is quoted from the NRDA´s home page:
"Rural areas make up more than half of Sweden. Distances from workplaces and various service outlets are long. Altogether the rural population of Sweden numbers some one and a half million people: an average of three and a half inhabitants per square kilometer. Of these, nearly half live in particularly sparsely populated areas. They are the main concern and focus of Glesbygdsverket, the Swedish National Rural Development Agency."

One of NRDA tasks is to define these rural areas and analyze the living conditions for the people living there. NRDA has partly done this by using a GIS system. The platform for doing these analyses has evolved during a 10-year period. The source data to this system are geographical datasets taken from the National Land survey (road networks) and Statistics Sweden (population statistics). Two methods have been used in this defining work. The first, fairly simple, is to look at the density of the population and the second one, a bit more complex, is to use proximity relations between the population and points of interest for the population e.g. food stores, train stations or hospitals. In this paper a design of a system of calculating the later will be presented.
Dealing with large detailed datasets is in line with NRDA´s task of making rural areas visible. If the datasets are generalized and lumped together, it is often on the expense of the rural areas.
In the early nineties the first generation of a proximity tool was developed. As the datasets became more detailed calculation times became a growing problem, despite the increased performance of computing power. In the year 2002 a decision was made to develop a new generation of the tool. Referenced in the paper as MapProx (*Map* for the mapping capabilities and *Prox* because it is a proximity tool).

When a new system is developed or bought by an organization the transparency of the system should be considered. Buying a ready made system in most cases tend to become a “black box” with few or no possibilities changing the system according to the demands of the organization, whereas a system developed inside the organization becomes a “transparent box”.
It seams to be a growing demand from organizations to get access to the source code for the applications they use. This is a point often brought forward in discussions in favor of Open Source concepts (Raymond, 2001).
Of course this discussion of trying to rank developing a new system with buying a pre-built system have more angles to it. In this project the transparency was an important factor when deciding to develop because of a legacy of failed projects with pre-built systems.
The MapProx tool is a one-purpose tool, namely performing proximity calculations. Tools with similar functions often have a multipurpose approach. They often tend to broaden their field of operation to please a waste amount of users. However, such designs often lead to compromises in the core function of the application. An example of this is the mixing of Proximity and Navigation. In a Proximity tool you are not interested in a description of the path from point A to point B you just want to know the travel distance or travel time. This fact has impact on the generalization process (discussed later) where a pure proximity calculation can use a more effective approach.
A good design is a design that can be implemented and maintained in a cost effective way. It is important for the designer to make himself understood by the developers. This is a limiting factor for the designer. It is often better to design an algorithm that is easy to understand than a complex algorithm of scientific excellence. With modern object oriented development methods it is also a good standard to start out with a simple algorithm and then change it to a more complex when you have a functioning overall system. The method of iterative development is supported by i.e. the Rational Unified Process (Kruchten, 2000)
Even in a relatively small project it is normal to have a group of developers who all must have the same picture of what’s going to be developed. During the evolution of software development a number of modeling languages been used. The one used in this project is Unified Modeling Language (Booch, Rumbaugh, and Jacobson, 1999).

Demands on the tool

The MapProx functionality has two implementations. Firstly it will work as a desktop application for the employees of NRDA, doing straightforward proximity analyses. Secondly it will work as a module in larger, more complex systems that have proximity analyses as a base function. An example of the later is a system for calculating tax adjustments between municipalities[1]. This means that the interface to the functions in the tool shall be accessed of both a human or another computer process.

The design of the tool should have a scalable approach. The source data used today within NRDA will with all certainty become larger, both expanding into new geographic areas (i.e. Europe) and become more detailed.

Doing proximity analysis is a core function within the NRDA, which makes it important to have a clear insight into the details of the application. The NRDA wants to have access to the source code to have this insight.

The design and the implementation shall consider the future trends in the computer hardware development aiming at i.e. computers with multiple processors and 64-bitsprocessors.

At a first glance the process of doing proximity analysis seems to be a fairly static task; ”once it’s done it’s done”. Considering changes in the source data and simulation functions dealing with iterations, high performance is an important demand on the application.

The development of the application should be done in such a way that dependencies to operating systems and GIS-platforms are held to a minimum. This allows the application to evolve into heterogeneous environments.

[1] The NRDA assists the Ministry of Finance to calculate tax adjustments dependent on structural differences between municipalities.

The Overall process

The overall process for doing calculating proximity with the MapProx application can be listed in the following sub processes.

- **Importing a road network and building topology**. This process imports a road network into the internal topological format.

- **Importing Start and Target points, connecting the points to the road network.** In this process the Start- and Target points are imported and connected to the road network.

- **The generalization process**. Simplifies the road network speed up calculations.

- **Calculating the distances.** The shortest distance between the start and target points are calculated.

- **Exporting the result**. The calculated data is added as an attribute to the start points and exported to a text file.

The internal topological format is serialized between the steps. If one of the steps goes wrong it is possible to restart at that sub process. The serialization process also makes one file for each of the Geographic Sub Areas. This makes it possible to do parallel calculations in a convenient way (discussed further in the discussion chapter).

When time consuming calculations are performed it is important to have interfaces for monitoring the progress of the calculations.

Importing a road network and building topology

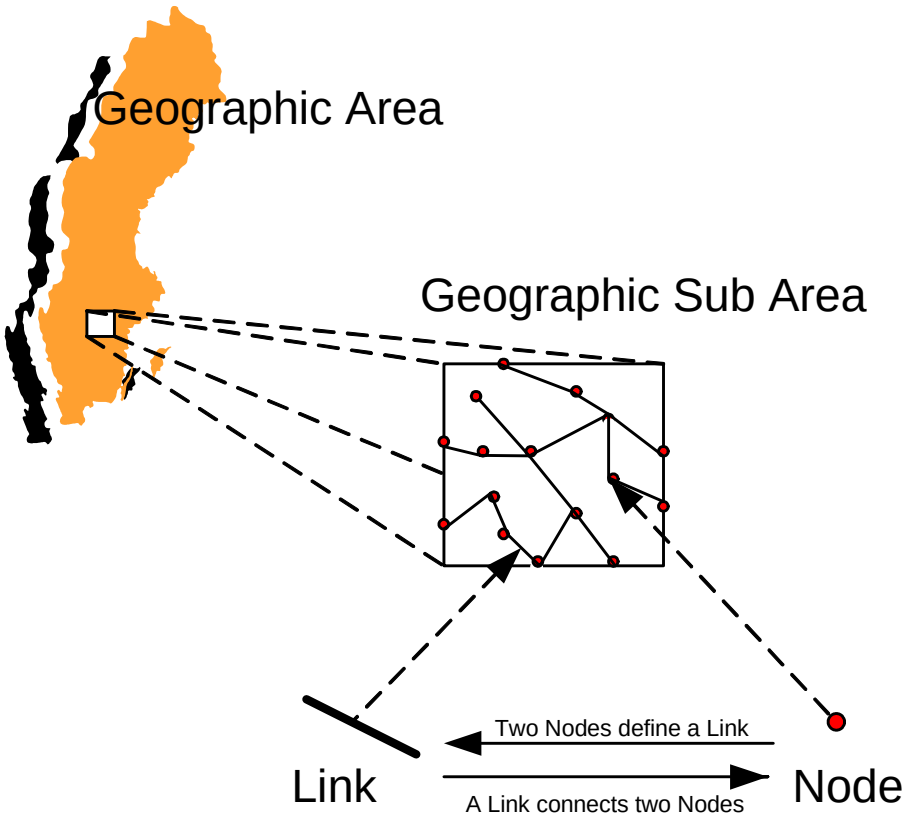

Figure 1. The figure describes a simplified schema of the topological format used.

Network source files to be imported into the system are not structured in a topological format (The geographic objects are not organized with geographic relations between them). When the road network is imported, the topology is built. Figure 1 shows the topological format the application uses in a simplified manner. There is a Geographic Area holding all instances of Sub Areas. One Sub Area holds the links and the nodes that lay within the area. The advantage with the approach of parting the Geographic Area into Sub Areas is that the Sub Areas are “independent” from each other, which make them easier to handle. The Sub Areas are serialized into individual files in the internal BNF-format. The Sub Areas can be handled independently in their own calculating threads.
The disadvantage with the parting is that a new type of nodes must be introduced on the borderlines between the sub areas. These border-nodes will appear in two adjacent sub areas. The intimate connections between Links and Nodes (in the implementation realized with pointers) are the core of the topological format.
As long as the road network stays intact the same imported network can be reused in forthcoming calculations.

Importing Start and Target points, connecting the points to the road network.

The Start and Target nodes can be delivered in different formats. The import function now accepts both MapInfo and ESRI formats. When Start and Target nodes are imported they are connected to the closest point on the road network. This can be a node or at right angles on a link. If the later method is used the link is broken up in two and a new node is introduced. Making this procedure an effective geographic indexation must be used. In MapProx a Kd-tree is used, e.g. (Berg, 2000).

The generalization process

When start points and target points are connected to the road network. The generalization process can begin. The goal with the process is to simplify the road network to make the calculations of the distances as smooth as possible without tempering with the correctness of the result.

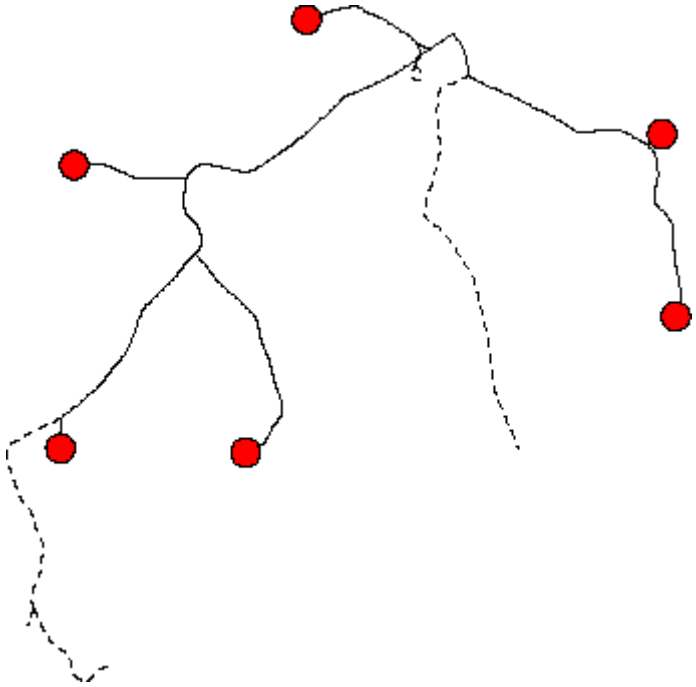

Figure 2. The figure describes removable links (dotted) in a road network. Red dots are start or target nodes.

One step is to remove all dangling poly lines that doesn't have of a start or target point. Figure 2 shows as dotted lines the poly lines that can be removed without tempering with the final result.

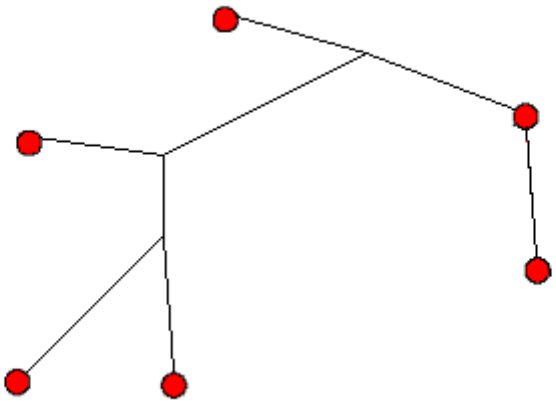

Figure 3. The figure shows how straight lines can replace complex poly lines.

Another generalizing step is to remove nodes in poly lines that are not start or target nodes, or have more than two links connected to it. When this is done it is important to recognize that the geometry of the link is destroyed by this method and the actual distance (or time-distance) is saved as an attribute to the link. This is shown in figure 3 where the road network from figure 2 is generalized in the described way.

Calculating the distances

Finding the shortest way in a network graph is a classic problem in computational and mathematic sciences. The algorithm used in this application is the Dijkstra algorithm described in many sources e.g.(Ahuja, Orlin, and Magnanti, 1993).
A function that is built into this step of the calculation is the possibility to use a categorization of the target points. This can be used to calculate the shortest distance to more than one type of target point in the same calculation.

Exporting the result

The result, distance and target point id, is delivered as attributes to the start points in a text file. Further analysis of the result can be done in a desktop GIS as MapInfo or ArcGIS.

Discussion of future design changes

The design is implemented in a functional application, but the work can also be looked at as a first step. The future development can now be done with the confidence that it is possible to go back one step and still have a working system. The object-oriented development methods allow the developers to look at the different parts of the application separate. The rest of the paper is a discussion about possible future development that is not implemented but will be of interest in future development.

Parallel processing

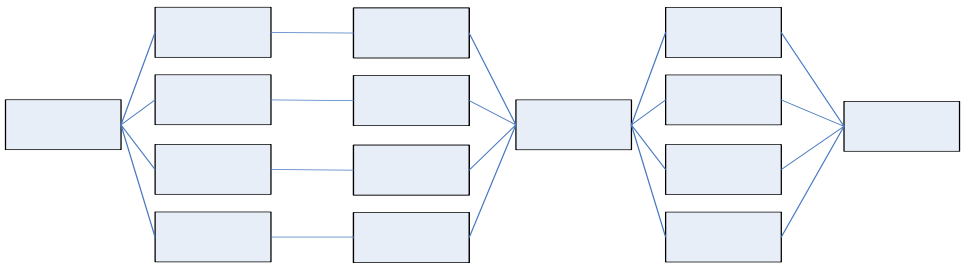

Figure 4. An overall process can consist of both single thread and multithread sub processes.

Doing time-consuming calculations every useful way of improving performance is of interest. It is possible to reach results by developing effective algorithms. Another way is to use fast computers. A third is to have more than one calculation, taking place at the same time.
If a parallel approach of using calculations is going to be used in the application it is important to realize this at an early stage of the design process.
The format of the data used in the tool should, if possible, be groped into self-contained sub sets. In this tool the Geographic sub areas are used as sub sets.
In the design it should be considered what processes that could use parallel calculations. One overall calculation process can use both a single thread approach in a sub calculation and a multithread approach in another (se figure 4). Examples of this can be the import of a single in-data file that is a single thread procedure. When the dataset is imported and parted into subsets, a multithread approach can be used.

Multiple Geographic Sub Areas interfaces

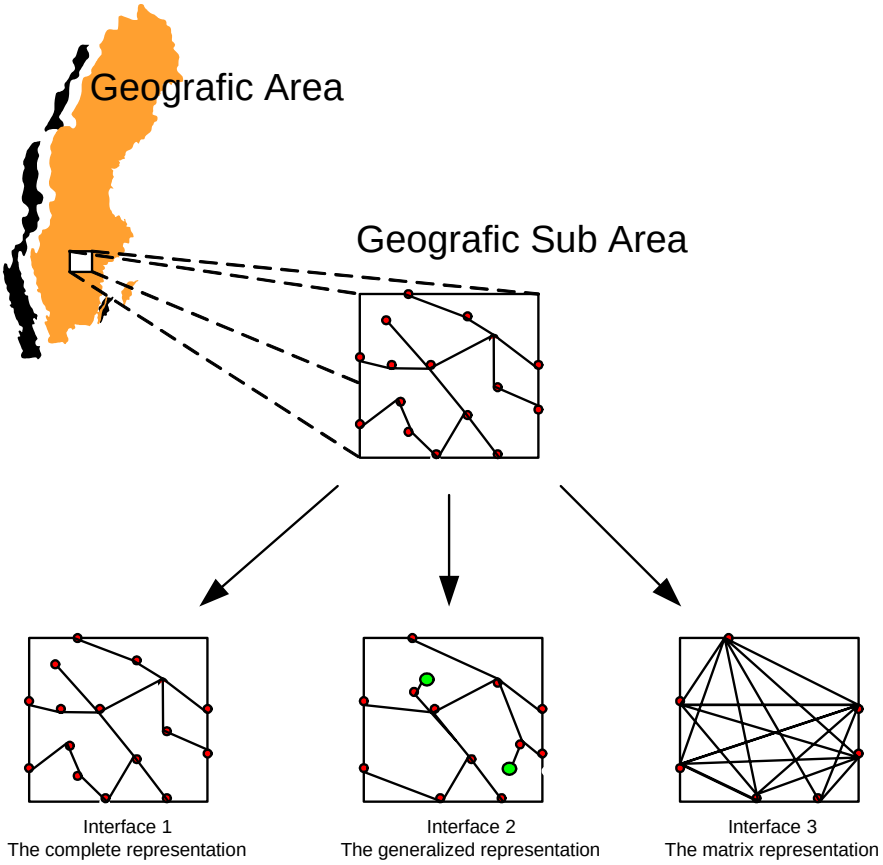

Figure 5 Three interfaces of the Geographic Sub Area component (The green dots in Interface 2 are start or target nodes)

The Geographic sub area could be developed into having a polymorphic behavior. The area should have three different interfaces (figure 5) used as described below. Perhaps this is more of a structural clarification than a big development step.

The processes that connect start and target points to the road network use the first interface. In this interface it is important that the right geographic representation of the road network is used.

When the start and target points are connected the generalization process can start presenting its result, a simplified road network, through the second interface. This is used for navigation within a Geographic sub area.

When a sub area only has the function as transportation between two other sub areas, i.e. the area does not have any start or target point of interest connected, a third interface can perhaps be of interest to implement. This interface shows the sub area as a matrix of distances between the sub areas border nodes. An interesting fact is that the matrix can be calculated before the connection of the start and target nodes.

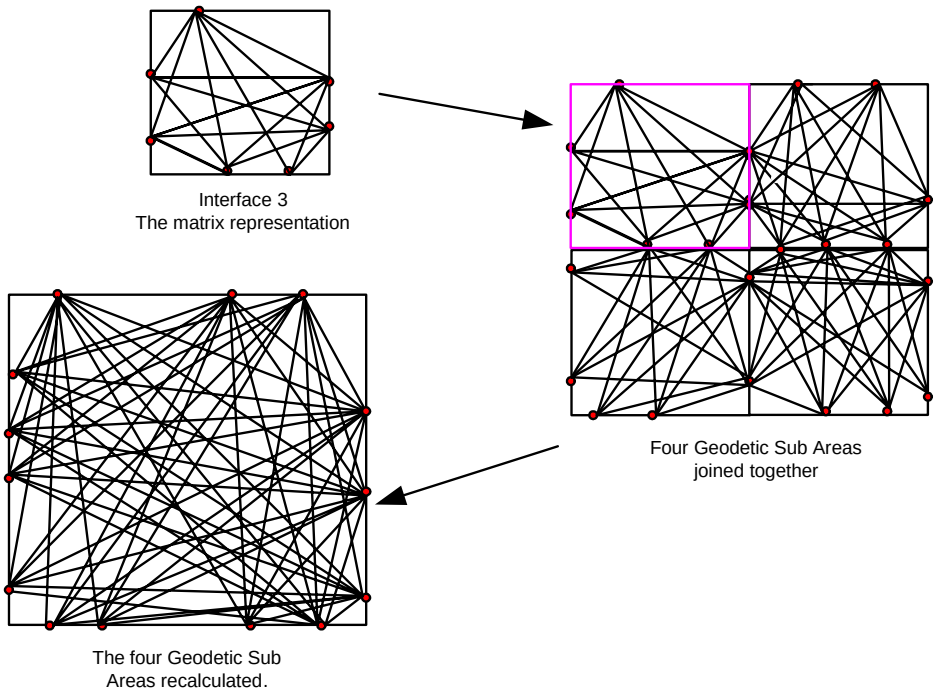

Figure 6. The figure shows how four Geographic Sub Areas can be bundled to one to speed up calculations.

To speed up long distance calculations further, sub areas can perhaps be sewn together to larger areas in a hierarchical layer structure only presenting large areas of the third type of interface to a user. This process can also be done before the connection of the start and target nodes as a pre step.

Small changes in source data.

When new source data is used in the application, there are often small changes from the datasets used in earlier calculations. In MapProx today a total recalculation is made regardless of the size or type of the changes. If there is a way of looking at the changes in the source data and what the impact of these changes are on the calculation, some incremental update routine could be implemented with a positive effect on performance.

References:

Ahuja, R. K., Orlin, J. B., and Magnanti, T. L. (1993). *Network flows : theory, algorithms and applications*. Prentice Hall, Englewood Cliffs, N.J.

Berg, M. d. (2000). *Computational geometry : algorithms and applications* (2., rev. ed.). Springer, Berlin.

Booch, G., Rumbaugh, J., and Jacobson, I. (1999). *The Unified Modeling Language user guide*. Addison-Wesley, Harlow.

Kruchten, P. (2000). *The rational unified process : an introduction* (2. ed.). Addison-Wesley, Reading, Mass.

Raymond, E. S. (2001). *The cathedral and the bazaar : musings on Linux and open source by an accidental revolutionary* (Rev. ed.). O'Reilly, Sebastopol, Calif.