Thesis Title: Efficient Spatial Query Processing Techniques

Abstract

With the increasing popularity of automated processes in fields like Earth Sciences, Cartography, Remote Sensing, and Land Information Systems, and the rapid increase in the availability of data from a wide variety of sources like satellite images, mapping agencies and simulation outputs, we are currently on the verge of a dramatic increase in both the quantity and the complexity of geo-spatial data sets. In the past ten years a great deal of research has been devoted to extending relational database systems to handle geo-spatial workload, but most of the research has focused on language issues and indexing techniques. This is unfortunate since the large data set problems are lurking beneath the surface and are now likely to surface with a vengeance. This thesis attempts to solve some of the challenges associated with empowering a database system to efficiently support geo-spatial query processing.

The first part of this thesis focuses on efficient algorithms for evaluating the spatial join. A spatial join involves combining, based on some spatial relationship, two spatial data sets to produce a third data set. For example map overlap, which requires combining two maps to produce a third, is a spatial join. This thesis presents a new algorithm called PBSM (Partition Based Spatial-Merge), that partitions the inputs into manageable chunks, and joins them using a computational geometry based plane-sweeping technique. PBSM can be thought of as the spatial equivalent of relational hash join algorithms. We have implemented PBSM in Paradise, a database system for handling geo-spatial data, and have compared the performance with other existing spatial join algorithms.

The second part of this thesis focuses on using parallelism to evaluate spatial queries. Two issues that are dealt with here are: how to decluster data in a parallel environment, and how to use parallelism to efficiently evaluate a spatial operation. Various spatial declustering strategies, that are based on partitioning the space in which the spatial objects lie and mapping the space partitions to nodes are presented. We have also developed various spatial join algorithms and are currently implementing them in the parallel version of Paradise, which currently runs on a cluster of PCs.

We have also developed algorithms for evaluating the spatial aggregate in a parallel environment. A spatial aggregate involves summarizing a spatial relationship over a group of spatial entities. An example of spatial aggregate is finding the “closest” spatial object to a specific point. While spatial aggregates are similar in spirit to the relational aggregate, there is an important difference here because a single spatial object may participate in multiple groups. For example, if we want to find the closest toxic waste dump to every city, each dump will probably have to be considered with respect to multiple cities. This in turn necessitates multi-step operators more complex that those used by parallel relational systems. We have developed algorithms for the spatial aggregate and are currently in the process of implementing them in Paradise.