

Abstract

An important problem in terrain analysis is modeling how water flows across a terrain and creates floods by filling up depressions. The accuracy of such modeling depends critically on the precision of the terrain data, and available high-resolution terrain models of even fairly small geographic regions often exceed the size of a computer's main memory. In such cases movement of data between main memory and external memory (such as disk) is often the bottleneck in the computation. Thus it is important to develop I/O-efficient modeling algorithms, that is, algorithms that minimize the movement of blocks of data between main memory and disk. In this paper we develop practically I/O-efficient algorithms for the problem of computing the areas of a terrain that are flooded in a given flash flood event due to water collecting in depressions. Previous work only considered events where rain falls at a constant uniform rate on the entire terrain. In reality, local extreme flash floods can affect downstream areas that do not receive heavy rainfall directly, so it is important to model such non-uniform events. Our main algorithm uses $O(\text{Sort}(N) + \text{Scan}(H \cdot X))$ I/Os, where N is the size of the terrain, $\text{Sort}(N)$ and $\text{Scan}(N)$ are the number of I/Os required to sort and read N elements in the standard two-level I/O-model, respectively, X is the number of sinks in the terrain and H the height of the so-called merge-tree, which is a hierarchical representation of the depressions of the terrain. Under practically realistic assumptions about the main memory size compared to X and H , we also develop $O(\text{Sort}(N))$ I/O-algorithms. One of these algorithms can handle an event in optimal $O(\text{Scan}(N))$ I/Os after using $O(\text{Sort}(N))$ I/Os on preprocessing the terrain. We have implemented our algorithms and show that they work very well in practice.