A Succinct Data Structure for Dynamic Queries

Pushkar Mishra
pushkarmishra@outlook.com
University of Cambridge

Kunal Kapila
kunalapila@gmail.com
Indian Institute of Technology, Kanpur

January 2015

Abstract

Segment trees are widely used to process queries like the Range Minimum Query (RMQ), the Range Sum Query (RSQ) and Stabbing Query. In this paper, we propose an isometric data structure, the "Segbit", to accomplish these tasks. Segbit is based on the principle that a range can be decomposed into smaller ranges, whose lengths are powers of 2. The operations to traverse ranges are based on binary representation of indexes. A Segbit performs queries and updates with the same time complexity as a Segment tree, but requires strictly less than $2 * N$ memory (which, unlike a segment tree, includes the space to store the original array on which Segbit is built), for an array of size N. It is shorter to code and easier to understand. The update/query routines of a Segbit are iterative and have a lower constant of complexity, thus being faster than their recursive counterparts of the Segment tree. Overall, the Segbit performs better than Segment trees.

Overview

Consider a one-dimensional array $A[1..N]$. We wish to answer a large number of dynamic Range Minimum Queries (RMQ) on arbitrary intervals of this array. By "dynamic", we mean that the intervals to be queried are specified during run-time and each query must be answered as and when it arrives. The queries are interspersed with update operations which require us to change the the element at a particular index to a new value. The RMQ can be replaced with other queries like the Range Sum Query, the Range Maximum Query, etc. Such a set of operations, where ranges are queried and individual elements are updated is called *Point-Update Range-Query*.

We also take up the dynamic "Stabbing Query" problem. Consider a set of intervals S. Each interval in S lies between the points 1 and N. For a point q, $S(q)$ is the number of intervals in S which contain the point q. The Stabbing Query problem asks for computing $S(q)$ efficiently. By "dynamic" we mean that intervals can be added to S, or can be deleted from it. Our implementation uses $O(N)$ memory. There are methods which utilize $O(s)$ memory, where s is cardinality of set S, however, we do not compare with those methods due to their complex implementations. This problem requires us to update ranges and query individual points, and hence, belongs to the category of *Range-Update Point-Query* problems. The above listed problems are of consequence to Computational Geometry, Graphs and Data Compression.

In 1977, J.L. Bentley, in his paper, presented a very powerful data structure known as the Segment tree, which can solve such queries in logarithmic time. The aim of this paper is to put forward a simple, comprehendible variant to the Segment tree. In practice, the Segbit is faster and uses significantly lesser memory than a segment tree. The memory and running time efficiency are more pronounced in multidimensional versions.