Compressing a Directed Massive Graph using Small World Model

Fang-Yie Leu
leufy@mail.thu.edu.tw

Yao-Chung Fan
g912810@student.thu.edu.tw

Computer Science and Information Engineering
Tung-Hai University, Taichung, Taiwan

Abstract

Representing a small graph is simple. Representing a graph with massive amount of vertexes and edges is an engineering challenge. In this article we propose a method that can compress a small-world-like massive digraph (directed graph) into at most half size of the original representation represented by using adjacency list. This method also provides a fast decompression algorithm that works as quickly as adjacency list does. Particularly, our approach is not mutually exclusive with other compression techniques. We can exploit a higher compression ratio by integrating some compression techniques, for example, Huffman coding and Delta coding, with our approach.

In this paper we deal with the problem of finding a compact representation of a graph from which the vertices adjacent to any specified vertex can be easily determined. The standard adjacency list approach consisting of two lists (one for incoming list and the other for outgoing) for each vertex is redundant since only incoming lists or outgoing lists are sufficient to reconstruct the graph. However, to eliminate one of the lists (incoming and outgoing) will dramatically great increase the complexity of determining the neighborhoods of a specified vertex. Exactly as this tradeoff problem, our work presents a compromise approach. First, we partition the graph into subgraphs such that most of the edges in the graph are contained within those subgraphs. Only few edges are cut. Second, the edges are classified into distant edges, those that are cut, and short edges, those in a subgraph. After that, the short edges in outgoing lists are eliminated and the distant edges in outgoing lists are retained. That is, the outgoing lists are now replaced by distant lists. Thus, the size of this representation will approach to half size of adjacency-list representation.

For determining what vertices are adjacent to a given vertex v, we can directly read the incoming list and distant list of v. The out-going list of v can be reconstructed by scanning incoming lists of the subgraph containing v. By this approach, we can acquire the entire linking information of v. Furthermore, via probabilistic analysis we can expect that the performance of finding all vertices adjacent to a given vertex of the proposed representation will work as quickly as adjacency-list one does.