Incremental Computations of Connectivity Queries in Sliding Windows over Streaming Graphs

Chao Zhang

David R. Cheriton School of Computer Science
University of Waterloo
chao.zhang@uwaterloo.ca

1 Introduction

Graphs have been the natural representation of data in many domains. With graph structured data, the most interesting operation is to compute *connected components* (CCs) [4], which are subsets of vertices in a undirected graph such that all vertices in the subset are connected via paths. Analyzing CCs has wide applications in practice, including social networks, transport networks, etc.

In modern data-driven applications, stream processing [5] is of significant importance. In stream processing, computations are typically applied in *sliding windows* [1] that are continuous finite subsets of streams over the infinite input stream. Sliding windows are defined using two parameters *range* and *slide*. For instance, a sliding window with range 3 hours and slide 2 minutes includes all the streaming data of the last 3 hours and the window is updated every 2 minutes, *i.e.*, deleting expired streaming data and inserting new streaming data.

The naive approach to compute sliding window connectivity is to traverse the streaming graph in each window instance of the sliding window, *e.g.*, performing breadth-first-search (BFS) in each window instance. Apparently, the naive approach cannot meet the requirement of real-time processing, which asks for high-through and low-latency computations. A non-trivial method is to use the well-known fully dynamic connectivity (FDC) data structures [2, 3]. Specifically, FDC supports 3 operations: `insert`, `delete`, and `query`. Obviously, the `insert` and `delete` operations supported by FDC can be used to deal with the updates required by sliding windows. The main bottleneck of the FDC approach is that the `delete` operation can have high latency as it requires traversing the entire graph in the worst case.

We design the bidirectional incremental computation (BIC) model to efficiently compute sliding window connectivity, which can reduce the problem of sliding window connectivity into a bidirectional computation. The main idea of BIC is that (i) streaming edges with contiguous timestamps are grouped to form disjoint chunks; (ii) window instances are split according to chunks; (iii) queries are processed by applying partial computations in chunks followed by merging the corresponding partial results. Specifically, we compute two kinds of buffers for each chunk: *forward* and *backward* buffers. Forward buffers are computed incrementally by scanning streaming edges *from the first to the last* in chunks while backward buffers are computed in the same way except that streaming edges are scanned *from the last to the first* in chunks. These two kinds of buffers are stored and merged to compute the query result of each window instance. Consequently, the overhead of performing the costly `delete` operation can be completely avoided.

In this presentation, we will elucidate the intricacies of incremental computations within the *forward* and *backward* buffers, as well as expound upon the merging operation. Our work is ongoing, and in addition to detailing our approach, we will also present preliminary experimental results in comparison to state-of-the-art methods based on FDC data structures.

References

[1] L. Golab and M. T. Özsu. Issues in data stream management. *ACM SIGMOD Rec.*, 32(2):5–14, jun 2003.

[2] M. R. Henzinger and V. King. Randomized fully dynamic graph algorithms with polylogarithmic time per operation. *J. ACM*, 46(4):502–516, 1999.

[3] J. Holm, K. de Lichtenberg, and M. Thorup. Polylogarithmic deterministic fully-dynamic algorithms for connectivity, minimum spanning tree, 2-edge, and biconnectivity. *J. ACM*, 48(4):723–760, 2001.

[4] S. Sahu, A. Mhedhbi, S. Salihoglu, J. Lin, and M. T. Özsu. The ubiquity of large graphs and surprising challenges of graph processing: extended survey. *VLDB J.*, 29(2):595–618, 2020.

[5] A. Toshniwal, S. Taneja, A. Shukla, K. Ramasamy, J. M. Patel, S. Kulkarni, J. Jackson, K. Gade, M. Fu, J. Donham, N. Bhagat, S. Mittal, and D. Ryaboy. Storm@twitter. In *Proc. ACM SIGMOD Int. Conf. on Management of Data*, page 147–156, 2014.