CS388G First Project Write-up

Stephen Roller
roller@cs.utexas.edu

October 17, 2011

Abstract

In this paper, we review the results of King and Sagert, 2002 [11]. King and Sagert present a novel, randomized, fully dynamic graph algorithm for maintaining the transitivity closure of a directed graph. Their algorithm can determine if two vertices $i$ and $j$ are connected in $O(1)$ time. Their algorithm is randomized and may falsely answer no with probability $O(1/n^c)$ for some constant $c$. Their algorithm has a worst case update time of $O(n^2)$ for acyclic graphs, and $O(n^{2+\omega})$ update time for general graphs, where $\omega \leq .26$. Additionally, we also review other related works, emphasizing the historical development of the problem, and briefly suggest avenues for further research.

1 Problem Overview

This paper explores work relating to fully dynamic graph algorithms for maintaining the transitivity closure. A fully dynamic graph algorithm provides a data structure for a graph and supports sequential updates to the graph. Updates may be in the form of insertions or deletions of edges or vertices. These algorithms differ from static graph algorithms, where a graph may not change after created, and from partially dynamic graph algorithms, where only a subset of edit-types are allowed.

The transitivity closure of a directed graph aims to answer queries of the form “is there a directed path from $u$ to $v$?” Fast answers to such queries have applications in many areas, including compilers and databases [11].

Before we begin a discussion of all the prior works, we wish to examine two extreme possibilities for solutions to this problem.

In the first case, we could approach the problem in the most naive fashion: we retain our tradition $O(1)$ time for insertion and deletion of edges (through the use of an adjacency matrix). Using this method, it’s easy to see this gives $O(n^2)$ worst case for queries [1].

In the other extreme, we could recompute the entire transitivity closure matrix after every single update. This yields a $O(1)$ query time, but a worst case $O(n^{2+\omega})$ update time, where $\omega$ is the exponent in the matrix multiplication problem [1].
Here we see a clear tradeoff: one can optimize for query time at the cost of update time, or vice versa. This tradeoff will repeatedly occur throughout the literature. In the rest of the paper, we present a (hopefully) comprehensive literature review of the fully dynamic directed graph transitivity closure problem, with emphasis on the role of King and Sagert 2002 [11] and its major contributions at its initial presentation.

2 Prior Work

The study of fully dynamic graph algorithms has a rich history extending some three decades [3]. The earliest partial dynamic algorithms for maintaining the transitive closure was Ibaraki and Katoh 1983 [8]. Their incremental algorithm (inserts only) had an update time of $O(n^3)$ over any sequence of insertions. The decremental algorithm (deletes only) had an update time of $O(n^2)$ per deletion. Both algorithms provided constant query time. While extremely inefficient, this was still the first success toward fully dynamic transitivity for directed graphs.

Later came La Poutre and van Leeuwen in 1998 [13], who developed another incremental dynamic algorithm, improving the bounds to $O(n)$ amortized update time with $O(1)$ query time. The decremental algorithm presented in the same paper provided an $O(m)$ cost per deletion with constant query time. This provided a significant improvement over Ibaraki and Katoh's work.

In 1993, Yellin [19] produced a fast incremental dynamic algorithm for maintaining the transitivity closure. It applied to bounded degree graphs and had a runtime of $O(m^*D)$ for $m$ edge insertions, where $m^*$ is the number of final edges in the graph and $D$ is the out-degree of the final graph. Additionally, Yellin proposed a decremental algorithm with $O(m^*D)$ running time for $m$ deletions, where $m^*$ is the initial number of edges. While the requirement of the bounded out-degree was limiting, this marked the first somewhat efficient implementations.

In 1995, Henzinger and King [7] presented a novel method for approaching the fully dynamic connectivity problem. Their method obtained an amortized bound of $O(p \log^2 n)$ for a series of $p$ updates, allowing for connectivity queries to be answered in worst case $O(\log n / \log \log n)$ time. Their technique combined novel graph decomposition with randomization, resulting in a Las Vegas type algorithm (one that always produces correct results). While this work applied only to undirected graphs, it marked a large step toward applying randomization techniques to the connectivity problem.

Also in 1995, Henzinger and King [6] presented an exponential improvement in the fully dynamic biconnectivity problem. Two vertices are said to be biconnected if and only if they are connected by two vertex-disjoint paths. Much like [7], this work only applies to undirected graphs.

However, in the same work, Henzinger and King also presented the very first fully dynamic algorithm for computing the transitive closure of a directed graph that performed better from recomputing from scratch. Their method obtained
a query time of $O(n/\log n)$ and update time of $O(m\sqrt{n} \log^2 n + n)$. Unlike their work in undirected graphs, this new algorithm was Monte Carlo; that is, it has a $O(1/n^c)$ probability of giving a false negative when answering “no”. This work marked a large breakthrough in the area, as previously only partially dynamic algorithms for directed graphs had been discovered at the time.

In 1997, Henzinger and King [5] presented a deterministic fully dynamic graph algorithm for maintaining a minimum spanning tree in $O(n^{1/3} \log n)$ amortized time per update operation. As an immediate consequence, they were able to answer the connectivity problem in $O(n^{1/3} \log n)$ per update operation and $O(1)$ worst case query time. This algorithm lowered the bounds for deterministic connectivity queries, but still could not perform as well as previous randomized algorithms. This work only applied to undirected graphs.

In 1996, Khanna, Motawani and Wilson [9] presented a new algorithm for maintaining the fully dynamic directed transitivity closure. Their work obtained a worst case constant query cost, but at the cost of some necessary lookahead. Specifically, they find that the fully dynamic transitivity closure can be maintained with $O((n\sqrt{mn})/\log n)$ amortized costs with $O(\sqrt{m/n} \log n)$ lookahead. They also show that using fast matrix multiplication techniques, fully dynamic maintenance is possible in amortized $O(n^{2.188}/\log n)$ cost per operation using $O(n^{0.188} \log n)$ lookahead.

3 Results of King and Sagert

In 1999, King and Sagert [11] presented a novel, fully dynamic graph algorithm for maintaining the transitivity closure of a directed graph. Their algorithm represented a large breakthrough in the field: it was the first such algorithm to allow for $O(1)$ worst case queries of connectivity without the use of lookahead queries.

This marked a significant improvement both on all prior work for fully dynamic directed connectivity algorithms. Compared to Khanna et al., this algorithm eliminated the need for any lookahead operations entirely. Compared to Henzinger and King, 1995, it reduced the query time from $O(n/\log n)$ to constant time.

Much like Henzinger and King, this algorithm is Monte Carlo; it has a one-sided probability of failure of $O(1/n^c)$, for any constant $c$. It is one-sided in that when a query returns “yes”, it is guaranteed correct. However, if a query answers “no”, then there is a polonomially low probability it is incorrect.

King and Sagert produce results for three cases: (1) acyclic graphs, (2) graphs with a bounded largest strongly connected component, (3) and general graphs. In all three cases, reachability queries of the form “Is there a directed path from vertex $u$ to vertex $v$?” in $O(1)$ time (with high probability of success as discussed earlier).

All bounds for graph edits (insertions or deletions) are all described in terms of inserting or deleting of edge sets. An edge set, $E_v$, is a set of incident edges to vertex $v$. The bound insertions or deletions of an edge set are given as:
1. Acyclic graphs: $O(n^2)$ worst case time.

2. Graphs such that $|SSC| \leq n^\epsilon$, where $|SSC|$ is the size of the largest strongly connected component: $O(n^{2+\epsilon})$.

3. General graphs: $O(n^{2.26})$ amortized time. The constant 2.26 comes from the fastest available rectangular matrix multiplication technique as of the time of their publication.

Additionally, their algorithm also supports sensitivity queries on acyclic graphs, or queries of the form “Is there a path connecting $u$ to $v$ without containing edge $e$?” in just $O(1)$ time.

King and Sagert also note that randomization is only used to reduce the wordsize used in computations. If a larger wordsize is allowed, then the algorithm becomes deterministic at the expense of computation time.

4 Recent Developments

The techniques of King and Sagert seemed to provide the necessary breakthroughs to further significant further research in the area.

Shortly after announcing her breakthrough work in 1999, King also announced a deterministic algorithm allowing for $O(1)$ lookup time and $O(n^2 \log n)$ worst case update time [10]. Unlike other works, this algorithm does not depend on fast matrix multiplication [1].

Here we note that worst case updates can possibly change $\Omega(n^2)$ entries in the transitive closure matrix, there is a natural $O(n^2)$ best bound for constant time connectivity queries. However, in 2000, Demetrescu and Italiano began to explore the tradeoff between query time and update time. At this point, they developed a deterministic algorithm for maintaining the dynamic transitivity closure matrix. In this work, they provide $O(n^2)$ amortized costs for updates while maintaining constant query time. In the partially dynamic case, an amortized cost of $O(n)$ was obtained [4].

From this work, they were able to develop a matrix-based approach that provided a direct tradeoff between query time and update time. In particular, they provided an algorithm for answering queries for acyclic graphs in $O(n^\epsilon)$ time with $O(n^{\omega-\epsilon} + n^{1+\epsilon})$ update time, where $0 \leq \epsilon \leq 1$ and $\omega$ is the $\epsilon$-dependent fast matrix multiplication exponent [2]. As of 2005, their best bounds were $O(n^{0.575})$ query time and $O(n^{1.575})$ update time. This new work provides a thorough understanding of the tradeoff between query time and update time. Like King and Sagert’s method, this Monte Carlo algorithm also has a one-sided probability of failure. This work, and many other contributions to dynamic fully directed graph algorithms, formed the basis of Demetrescu’s dissertation in 2001 [1].

Later in 2001, King presented a simple space saving trick to reduce the space complexity of various dynamic directed graphs [12]. In particular, she reduced
the space complexity of [4] from $O(n^3)$ to $O(n^2)$, while preserving the amortized time bound.

In 2002, Roditty and Zwick furthered the field by presenting an extensive paper providing many results in terms of the number of edges, rather than the number of vertices as in previous works. For fully dynamic graphs, they provide (1) a deterministic algorithm with amortized $O(m\sqrt{n})$ update time and worst case query time of $O(\sqrt{n})$ and (2) a randomized algorithm with amortized $O(m^{0.58}n)$ update time and worst case query time of $O(m^{0.43})$. Finally, when limiting their scope to acyclic graphs, they provide a deterministic algorithm with $O(m)$ worst case insert time, $O(1)$ amortized delete time and worst case query time of $O(n/\log n)$ [17]. Their work samples choice techniques of many previous works.

Roddity 2003 [15] obtained a novel algorithm whose primary appeal was its simplicity. It provided a $O(mn + pn^2)$ total running time, where $p$ is the total number of insertions and deletions, with queries answered in constant time. In addition to providing answers to connectivity, their algorithm can provide the path between two connected vertices in time proportional to the length of the path.

In 2004, Roditty and Zwick [16] presented a deterministic algorithm which reduced the bounds of their previous work to $O(m + n \log n)$ amortized update time and worst case query time of $O(n)$. This algorithm applies to general directed graphs and is the first work to break the $O(n^2)$ update barrier for graphs with $o(n^2)$ edges. Once again, we see a clear tradeoff between update time and query time: their work provides the fastest update time for the general case of recent work, but the slowest update time. Additionally, they provide a novel algorithm for maintaining the strongly connected components of a graph in $O(m\alpha(m, n))$ time, where $\alpha(m, n)$ is the Inverse Ackermann function seen in the Union-Find data structure.

The most recent works in the field have returned to studying how the fully dynamic transitivity closure is affected by the availability of lookahead during updates. One such work is Patrascu and Thorup 2007 [14]. They use a framework in the spirit of the Union-Find problem to provide an algorithm for $O(p \lg^2 n \lg \lg n)$ update time and $O(\lg \lg n)$ query time given $p$ lookahead operations.

The other recent paper taking advantage of lookahead operations is Sankowski and Mucha 2006 [18], which provides a randomized, one-sided error algorithm with query and update cost of $O(n^{2-\omega})$ time, where $\omega$ is the exponent of matrix multiplication. They also present a modified version with faster amortized queries, as well as algorithms with restricted types of updates based on matrix transformations.

Perhaps the last recent work worth noting is Demetrescu and Italiano 2006 [3], which provides an introductory literature review for those new to fully dynamic directed graph algorithms. In addition to providing a thorough literature review, they aim to bring together methods from recent years into a unifying framework.
5 Possible Further Research

There are many possible avenues of further research. Since many fully dynamic graph algorithms may be expressed in terms of each other, it would be interesting to see how recent work by Roddity, which expresses updates in terms of number of initial edges, could potentially lower bounds in other dynamic graph algorithms.

While efficient algorithms already exist for maintaining the transitivity closure of undirected graphs, it would also be interesting to see how any number of these algorithms perform on undirected graphs, and where special properties of undirected graphs would come into play.

Finally, given the recent return to analysis of lookahead-enabled algorithms, we also suggest this as a possible area of further research. In particular, the relationship of [14] to the Union-Find problem indicates the potential for application of the alpha technique.

6 References Notes

In the following list of references, we star (*) papers which appear cited (either in journal or conference form) in King and Sagert. All other papers can be assumed to be either prior works not cited in King and Sagert, or more recent works since. Refer to the sections above for a breakdown.

References


