Engineering shortest-path algorithms for dynamic networks

Mattia D'Emidio and Daniele Frigioni

Department of Information Engineering, Computer Science and Mathematics, University of L'Aquila, Via Gronchi 18, I–67100, L'Aquila, Italy.
mattia.demidio@univaq.it, daniele.frigioni@univaq.it

Abstract. The problem of updating shortest paths in networks whose topology dynamically changes over time is a core functionality of many nowadays networked systems. In fact, the problem finds application in many real-world scenarios such as Internet routing and route planning in road networks. In these scenarios, shortest-path data are stored in different ways and have to be updated whenever the underlying graph, representing the network, undergoes dynamic updates. This paper provides a top-level overview of [13], where new dynamic shortest-path algorithms for various real-world applications are proposed, engineered, analyzed and compared to the literature, both theoretically and experimentally.

1 Introduction

In recent years, a massive interest has arisen in the scientific community for new algorithms explicitly designed for nowadays computing systems, such as computer networks, transportation infrastructures and distributed systems. This impulse was motivated by the increasing complexity of such systems, which required new methods and solutions able to overcome the limits of purely theoretical and mathematical approaches to solve problems. In fact, both increasing demand for more efficient solutions to actual real-world problems and advancements in computer hardware, which render traditional computing models more and more unrealistic, have led to a rising gap between classical algorithm theory and algorithmics in practice. The emerging discipline of *algorithm engineering* aims at bridging this gap by complementing theory by the benefits of experimentation. This area of studies has gained even more importance in the last decade, when networked, therefore complex and in most cases dynamic, systems have undergone an astonishing widespread diffusion (see [5, 12, 18, 21]).

In [13] we have focused on engineering new algorithms for the dynamic single-source shortest paths problem, i.e. the problem of computing and updating shortest-path trees in networks whose topology dynamically changes over time. The study was motivated by the importance of this problem, which finds application in many real-world scenarios, such as routing in communication networks and route planning in road networks. In these scenarios, shortest-path data are stored in different ways and need to be updated whenever the underlying graph undergoes dynamic updates. In details, the original contribution of [13] consists of new dynamic shortest-path algorithms for various real-world applications. The work concentrates on problems related to three main categories of networks: *General Networks*, *Communication Networks*, and *Transportation Networks*, respectively. The proposed algorithms are analyzed and compared to the literature, both theoretically and experimentally. In Sections 2–4 we summarize the contributions for each of the aforesaid categories, while Section 5 gives some concluding remarks.

2 General Networks

In this part, we focused on the problem of maintaining the shortest-path tree from a given source of a general graph with positive real edge weights, whose topology undergoes dynamic changes. This problem has been widely studied both theoretically and experimentally. From the theoretical point of view, some solutions have been proposed [14, 15, 17, 20]. Some of them are only able to cope with the update of one edge at a time [14, 15], while others can handle also *batch* updates [17, 20], i.e., updates that consist of multiple edge changes at a time. To the best of our knowledge, none of the above solutions is asymptotically better than recomputing the shortest paths from scratch, by applying Dijkstra's algorithm in the worst case. From the experimental point of view, very few studies are known. The most recent is that in [2], an experimental evaluation of the algorithms in [15, 17, 19, 20] and some of their variants for batch updates. The most important conclusion of this paper is the astonishing level of data dependency within the problem. The second outcome is that it is useful to process a set of updates as a batch when updated edges have strong interference w.r.t. their impact on the shortest-path tree. While updates that are far away from each other usually do not interfere, and hence they can be handled iteratively.

Our contribution to this area is the following: we have developed two new dynamic algorithms for *homogeneous* batches [6], i.e. either *incremental* (containing only insert and weight decrease operations) or *decremental* (containing only delete and weight increase operations) batches which model realistic dynamic scenarios like node failures in communication networks. We have showed that they extend the results of [14] to general graphs, and to batch updates, and those of [15] to batch updates. We have proved the new algorithms to be theoretically efficient in case of homogeneous batches. We have also provided an extensive experimental study that compares the new solutions with the most effective known batch algorithms [7]. Our data show that the proposed algorithms improve over the literature in a set of realistic scenarios. Our results complement previous studies and show that the various solutions can be consistently ranked on the basis of the type of homogeneous batch and of the underlying network.

3 Communication Networks

In this part, we considered the problem of routing in communication networks. The most used approach for solving this problem is that based on shortest paths [4, 16]. If the network is represented by a weighted graph, where vertices model nodes of the network, edges model links connecting such nodes and the weight of an edge models the time required by packets for traversing the corresponding link, the problem can be solved by the distributed computation of all-pairs shortest paths. Known solutions for the problem are usually classified as *distance-vector* and *link-state* [3]. Most distance-vector solutions are based on the classical distributed Bellman-Ford's method and hence converge very slowly, due to well known looping phenomena, but require to store very little data and are able to broadcast information about dynamic changes to a subset of nodes of the network. Link-state algorithms require nodes to store the entire network topology and to compute the shortest path to any destination, usually by running Dijkstra's algorithm, thus requiring quadratic space per node. Link-state algorithms are free of looping, however each node needs to receive and store up-to-date information on the entire network topology after a dynamic change.

In the last decade, there has been a renewed interest in devising new lightweight distributed shortest-path algorithms for a large number of applications where routing devices can have limited storage capabilities, like i.e. wireless sensor networks and large scale ethernet networks. In these applications, loop-free distance-vector algorithms seem to be an attractive alternative to link-state algorithms [21].

Our contribution to this area is twofold. First, we have presented a new loop-free distance-vector algorithm, named *LFR* (Loop Free Routing), which improves over previously known algorithms [9]. From the theoretical point of view, the algorithm has the same message complexity of DUAL [16], one of the best algorithms of the category, but it is always the best choice in terms of memory requirements. From the experimental point of view, LFR outperforms DUAL [9] in terms of messages on a set of real-world networks, whereas DUAL is always the best choice on artificial instances. Second, we have developed a new technique, named *DCP* (Distributed Computation Pruning), which can be combined with every distance-vector algorithm to overcome some of their main limitations (high number of messages sent, high space occupancy per node, low scalability, poor convergence) in power-law networks [10]. We have provided experimental evidences that the use of DCP in combination with DUAL and LFR induces a massive improvement in their performance, in terms of both message complexity and memory requirements.

4 Tranportation Networks

In this part, we studied the problem of computing best connections in transportation networks. The main effort of the study was dedicated to best connections in road graphs, where vertices represent points on a map, edges represent road segments connecting such points, and travel times for each segment are assigned to the corresponding edge. This problem is a variant of the single-source shortest paths problem and hence can be solved by applying Dijkstra's algorithm. Unfortunately, real-world transportation networks tend in general to be huge, yielding unsustainable times to compute shortest paths by traditional approaches. For this reason, many efforts have been done in the last years to accelerate the practical performance of Dijkstra's algorithm on typical instances of road networks. These research efforts have led to the development of a number of so-called *speed-up techniques*, which compute additional data in a preprocessing phase in order to accelerate the answer to shortest-path queries in an on-line phase. Theoretically, none of such speed-up techniques is better than Dijkstra's in the worst case, while, in practice, some of them have been shown to be very effective. For a comprehensive survey we refer to [1].

The main drawback of these techniques is that, in general, they do not work well in dynamic scenarios, when edge weight changes occur to the network due to, e.g., traffic jams. These scenarios are, of course, interesting, as they arise frequently in practice. In order to keep shortest-path queries correct, the preprocessed data need to be updated. The easiest way is to recompute the data from scratch after each change. This is in general unfeasible, as even the fastest methods need too much time. Therefore, in recent years some techniques for updating shortest paths in dynamic scenarios [11, 22] have been developed.

In our work, we focused on the speed-up technique named Arc-Flags and proposed a new approach to efficiently use Arc-Flags in dynamic networks [8]. The new approach consists of a new data structure and a new fully dynamic algorithm. Our study was motivated by the fact that some of the best performing speed-up techniques, such as for example, CHASE and TNR+AF [1], rely on the correctness and the performance of Arc-Flags. Hence, our dynamization of Arc-Flags represents a first step toward the dynamization of these techniques. We provided both theoretical and experimental evidences that confirm this statement. In detail, our study shows that our dynamic approach overcomes previous methods for maintaining the Arc-Flags in dynamic networks.

5 Conclusion

We have proposed a top-level overview of [13], which contains novel contributions in the area of algorithm engineering. In details, we have given, analyzed and compared to the literature, new dynamic shortest-path algorithms for various real-world applications. The new algorithms improve over known approaches in many interesting scenarios and represent a step forward in the development of more efficient shortest-path solutions for dynamic networks.

References

1. R. Bauer, D. Delling, P. Sanders, D. Schieferdecker, D. Schultes, and D. Wagner. Combining hierarchical and goal-directed speed-up techniques for Dijkstra's algorithm. *ACM Journal on Experimental Algorithmics*, 15:Article 2.3, 2010.
2. R. Bauer and D. Wagner. Batch dynamic single-source shortest-path algorithms: An experimental study. In *8th International Symposium on Experimental Algorithms (SEA 2009)*, volume 5526 of *Lecture Notes in Computer Science*, pages 51–62. Springer, 2009.

3. D. Bertsekas and R. Gallager. *Data Networks.* Prentice Hall International, 1992.
4. S. Cicerone, G. D'Angelo, G. Di Stefano, and D. Frigioni. Partially dynamic efficient algorithms for distributed shortest paths. *Theoretical Computer Science*, 411:1013–1037, 2010.
5. S. Cicerone, G. D'Angelo, G. Di Stefano, D. Frigioni, and V. Maurizio. Engineering a new algorithm for distributed shortest paths on dynamic networks. *Algorithmica*, 66(1):51–86, 2013.
6. A. D'Andrea, M. D'Emidio, D. Frigioni, S. Leucci, and G. Proietti. Dynamically maintaining shortest path trees under batches of updates. In *20th International Colloquium on Structural Information and Communication Complexity (SIROCCO 2013)*, volume 8179, pages 286–297. Springer, 2013.
7. A. D'Andrea, M. D'Emidio, D. Frigioni, S. Leucci, and G. Proietti. Experimental evaluation of dynamic shortest path tree algorithms on homogeneous batches. In *13th International Symposium on Experimental Algorithms*, volume 8504 of *Lecture Notes in Computer Science*, pages 283–294. Springer, 2014.
8. G. D'Angelo, M. D'Emidio, and D. Frigioni. Fully dynamic update of Arc-Flags. *Networks*, (63):283–294, 2014.
9. G. D'Angelo, M. D'Emidio, and D. Frigioni. A loop-free shortest-path routing algorithm for dynamic networks. *Theoretical Computer Science*, 516:1–19, 2014.
10. G. D'Angelo, M. D'Emidio, D. Frigioni, and D. Romano. Enhancing the computation of distributed shortest paths on real dynamic networks. In *1st Mediterranean Conference on Algorithms (MEDALG 2012)*, volume 7659 of *Lecture Notes in Computer Science*, pages 148–158, 2012.
11. D. Delling, A. V. Goldberg, T. Pajor, and R. F. F. Werneck. Customizable route planning. In *10th International Symposium on Experimental Algorithms (SEA 2011)*, volume 6630 of *Lecture Notes in Computer Science*, pages 376–387, 2011.
12. D. Delling, P. Sanders, D. Schultes, and D. Wagner. Engineering Route Planning Algorithms. In *Algorithmics of Large and Complex Networks*, volume 5515 of *Lecture Notes in Computer Science*, pages 117–139. Springer, 2009.
13. M. D'Emidio. Engineering shortest-path algorithms for dynamic networks. Ph.D. Thesis, University of L'Aquila, Advisor: Prof. Daniele Frigioni, 2014.
14. D. Frigioni, A. Marchetti-Spaccamela, and U. Nanni. Semi-dynamic algorithms for maintaining single source shortest path trees. *Algorithmica*, 22(3):250–274, 1998.
15. D. Frigioni, A. Marchetti-Spaccamela, and U. Nanni. Fully dynamic algorithms for maintaining shortest paths trees. *Journal of Algorithms*, 34(2):251–281, 2000.
16. J. J. Garcia-Lunes-Aceves. Loop-free routing using diffusing computations. *IEEE/ACM Trans. on Networking*, 1(1):130–141, 1993.
17. P. Narváez, K.-Y. Siu, and H.-Y. Tzeng. New dynamic algorithms for shortest path tree computation. *IEEE/ACM Trans. on Networking*, 8(6):734–746, 2000.
18. E. Pyrga, F. Schulz, D. Wagner, and C. Zaroliagis. Efficient models for timetable information in public transportation systems. *ACM Journal of Experimental Algorithmics*, 12:2.4:1–2.4:39, 2008.
19. G. Ramalingam and T. Reps. On the computational complexity of dynamic graph problem. *Theoretical Computer Science*, 158:233–277, 1996.
20. G. Ramalingam and T. W. Reps. An incremental algorithm for a generalization of the shortest paths problem. *Journal of Algorithms*, 21:267–305, 1996.
21. S. Ray, R. Guérin, K.-W. Kwong, and R. Sofia. Always acyclic distributed path computation. *IEEE/ACM Trans. on Networking*, 18(1):307–319, 2010.
22. P. Sanders and D. Schultes. Dynamic Highway-Node Routing. In *Proc. of the 6th Workshop on Experimental Algorithms (WEA'07)*, volume 4525 of *Lecture Notes in Computer Science*, pages 66–79, 2007.