SEARCH IN DYNAMIC GRAPHS BASED ON CROWDSOURCING

Author: Truhin Alexandr
Supervisor: Railean Alexandr

Technical University of Moldova
Email: bmbalex@gmail.com, ralienpp@gmail.com

Abstract: *Applications that are working on routing rely on graph theory. But in today's world, data is changing very fast, and there is a need in having real-time or near-real-time results. This article presents a way of obtaining cheap near-real-time environmental data for route searching, and reviews several algorithms that are able to process such dynamic data. The need of different graph search algorithms is dictated by the fact that producing optimal results may be infeasible for real life use-cases, the fact that environment models are usually imperfect, and that environments are dynamic.*

Keywords: *crowdsourcing, dynamic graphs, dynamic graphs search, Anytime D*.*

1. Introduction

Today, routing applications rely on static data, preset in their memory. But as we live in a changing world, the future is all about real-time data and real-time results. There are some services that are collecting real time data, processing them and providing real-time results about roads and traffic congestions [1]. But such services are using real time data for updating their databases periodically, but not in real-time. Such services are just providing information about routes' states, without filtering results based on best cases (ex. less congested traffic routes). This article describes how to use real-time data for providing real-time results, by taking into account two facts:

- producing an optimal plan may be infeasible for real life use-cases, so we have to use near-optimal solutions;
- models are usually imperfect, and environments are dynamic, so there is a need in a solution that will solve these constraints while avoiding very frequent result updates.

2. Crowdsourcing

Crowdsourcing is the phenomenon in which the old arrangement, where only a few people produce content and many others use it - is reversed. Crowdsourcing represents outsourcing the production process to the masses, which can publish their content without applying traditional quality filters over it [2].

Crowdsourcing for routing services is represented by the information that comes from people who are using these services. The received data may be different, depending on which information is being used by the service for generating its planning results, the most common information is: availability of roads, precisions of roads positions, blocked roads, average speed on some portions of the roads, etc.

3. Graph search

The two most commonly used algorithms for searching an optimal path through graphs are: A* search algorithm [3] and Dijkstra's algorithm [4]. The efficiency of these algorithms is based on the facts that they process the minimum number of states possible, providing an optimal path solution when graphs are static. But the complexity of the algorithms increases as graphs are growing in size, so even these efficient algorithms may need some time to compute the optimal solution. Another edge-case may occur when after finding the optimal solution, the environment may change, so the user may be unable to follow the initially proposed solution. That's why there will be a need in updating the solution, and for most accurate results - the solution should be updated all the time. All these recomputations will cause delays in following the route. So there is a need in alternative ways of solving the problem.

4. Anytime algorithms

Anytime planning algorithms are based on the idea of finding the best results in the available time [5]. They have the ability to provide suboptimality bounds on the quality of the solution at any point of time. They start by quickly finding an approximate suboptimal solution, spending the remaining time for improving the result until finding the best one, if there is enough time. They are good in cases when we have to provide at least approximate results as fast as possible, and after that - using the rest of the time to deliver a more accurate and optimal solution. If an anytime algorithm returned an initial suboptimal result, we may stop its execution if there is no need to refine it (if environment changed), still having an approximate result.

5. Incremental algorithms

Anytime algorithms are good in cases when environment models are known a priori, and are less accurate when the environment is dynamic. This happens because of the need in frequent world model updating and recomputing the results. In such cases, a good solution will be to use incremental algorithms (also named repairing algorithms) that are using the results of previous planning to generate new results if the problem changed slightly. One such algorithm is D* [6]. Initially this algorithm is performing A* search to generate an initial solution, then, when the world model is updated, it repairs its previous solution using as much as possible from previous computations. This is giving a huge boost in many cases where the world model is changing. But it has a drawback if we take into account the proprieties of anytime algorithms: as soon as it finds any solution – it is stopping instead of searching for a better one.

6. Anytime and incremental algorithms

Anytime algorithms provide fast feedback and keep searching for optimal solutions for as much time they have; they deal poorly with computing frequently changing models. Incremental algorithms are opposed to that, giving the flexibility of reusing things that were already computed, but failing in case of limited time or in case of need to get one more accurate result after finding a general one.

The algorithm that is able to overcome these problems is Anytime D* [7]. It performs a search within given time constraints, but it is also able to reuse previous planning efforts in dynamic domains.

Conclusion

For providing a competitive and qualitative routing service in our days, it is not enough to just give rough planning results to the user, there is also a need in fast-responding services, with optimal results and flexibility for dynamic changes of the environment. All these things are possible, and one cheap and effective solution to this problem is to use crowdsourcing along with anytime and incremental algorithms.

Even in cases where there is no need in a fast-responding and flexible service, merely using anytime algorithms may boost the service performance and lower the prices for computational units, as users do not always need the most optimal result, or the most detailed one, they just need a solution.

Bibliography

14. Яндекс, Как работают Яндекс.Пробки (how do Yandex traffic congestions works), Retrieved 2011.12.05 - http://company.yandex.ru/technology/yaprobki/
15. JIS, A. Huberman, M. Romero, Fang W, Crowdsourcing, attention and productivity, Retrieved 2011.12.05 - http://www.hpl.hp.com/research/scl/papers/crowd/crowd.pdf
16. P. E. Hart, N. J. Nilsson, B. Raphael, A formal basis for the heuristic determination of minimum cost paths, IEEE Transactions on Systems, Science, and Cybernetics SSC-4 (2) (1968) 100–107.
17. Dijkstra, A note on two problems in connexion with graphs, Numerische Mathematik 1 (1959) 269–271
18. S. Zilberstein, S. Russell, Approximate reasoning using anytime algorithms, in: Imprecise and Approximate Computation, Kluwer Academic Publishers, 1995
19. Stentz, Anthony (1994), "Optimal and Efficient Path Planning for Partially-Known Environments", Proceedings of the International Conference on Robotics and Automation: 3310–3317
20. M. Likhachev, D. Ferguson, G. Gordon, A. Stentz, S. Thrun, Anytime Search in Dynamic Graphs, 23 October 2007, Retrieved 2011.12.05 - http://www.cs.cmu.edu/~maxim/files/ad_aij08_preprint.pdf