Fast point-to-point shortest path queries on dynamic road networks with interval data

Giacomo Nannicini [1,2] Philippe Baptiste [1] Daniel Krob [1]
Leo Liberti [1]

Key words: Shortest path, dynamic arc costs, PTAS, real-time data

1 Introduction

Consider a weighted directed graph $G = (V, A, c)$ (where $c : A \to \mathbb{R}_+$) which represents a road network evaluated by travelling times (so the graph may not be Euclidean). Assuming that c changes every few minutes, that the cardinality of V is very large (several million vertices), and that each shortest path request must be answered in a few milliseconds, current technology does not allow us to find an exact optimum in the brief time window before the next change of the weight function. Such a situation occurs in fastest path computations for GPS enabled vehicles with real-time traffic information. Some practically efficient algorithms for graphs with fixed arc costs are [SS05,GKW05]; [MG04,MGD04,SP06] address uncertainty in arc costs but do not yield algorithms which perform within acceptable time frames.

We assume some lower and upper bounding functions $l, u : A \to \mathbb{R}$ for c are known. In this paper, we propose a Polynomial-Time Approximation Scheme (PTAS) for the Point-to-Point Shortest Path Problem (PPSPP). The stringent time constraints do not make exact algorithms an acceptable choice, yet a guarantee on the solution quality is desired. Our algorithm is based on Dijkstra-type searches performed on clusters of nodes; such clusters are precomputed in such a way as to give a bound on the solution performance, whilst

Email addresses: giacomo.nannicini@v-trafic.com (Giacomo Nannicini), baptiste@lix.polytechnique.fr (Philippe Baptiste), dk@lix.polytechnique.fr (Daniel Krob), liberti@lix.polytechnique.fr (Leo Liberti).
[1] LIX, École Polytechnique, 91128 Palaiseau, France
[2] Mediamobile, 10 rue d'Oradour sur Glane, Paris, France

accelerating the search enough to be practically useful within the given time constraints.

2 Guarantee regions

For $s, t \in V$ we denote the set of all paths $(s, \ldots, t)$ from s to t by $P(s,t)$ and the set of all shortest paths from s to t on a graph weighted by function f by $P_f^*(s,t)$ (the subscript f is omitted when $f = c$). Given $U \subseteq V$ such that $s, t \in U$, let $G[U]$ be the subgraph of G induced by U. The set of all paths between s and t in $G[U]$ is denoted by $P[U](s,t)$ and the set of all shortest paths between s and t in $G[U]$ weighted by function f is denoted by $P_f^*[U](s,t)$; as before, we will write $P^*[U](s,t) = P_c^*[U](s,t)$. We naturally extend c to be defined on paths $p = (v_1, \ldots, v_k)$ by $c(p) = \sum_{i=1}^{k-1} c(v_i, v_{i+1})$.

Let $G_l = (V, A, l), G_u = (V, A, u)$ be the graph G weighted by the lower and upper bounding functions l, u. For $K > 1$, $s, t \in V$ and path $p \in P_u^*(s,t)$ (p is a shortest path from s to t in G_u), we define a *guarantee region* $\Gamma_{st}(K,p) = \{v \in V | v \in p \vee \exists\, q \in P(s,t)\ (v \in q \wedge l(q) < \frac{1}{K} u(p))\}$. We can prove that such a set of nodes has the following approximation property: $p^* \in P^*(s,t)$ and $q^* \in P^*[\Gamma_{st}(K,p)](s,t)$, we have $c(q^*) \leq K c(p^*)$. Although guarantee regions generated by shortest paths in G_u may not be the minimally-sized sets with this approximation property, it is possible to show that they are no worse than those generated by any other path.

The trouble with the guarantee regions defined above is that building all guarantee regions for all node pairs in a very large graph is not a feasible task with current technology. We deal with this problem by covering V with clusters $V_1, \ldots, V_k$; however, not all coverings are useful for our approach. We will call a covering $V_1, \ldots, V_k$ of V *valid* if for all $i \leq k$ there are two selected (not necessarily distinct) vertices $s_i, t_i \in V_i$ such that for all other vertices $v \in V_i$ there are paths $p \in P(v, s_i), q \in P(t_i, v)$ entirely contained in V_i. For all $i \leq k$ let $\sigma_i = \max_{v \in V_i, p \in P_u^*(v, s_i)} c(p)$ and $\tau_i = \max_{v \in V_i, p \in P_u^*(t_i, v)} c(p))$ be the costs of the longest shortest path in G_u from v to s_i and respectively from t_i to v over all $v \in V_i$. We can now extend guarantee regions to depend on a source and a destination cluster in a valid covering $V_1, \ldots, V_k$ of V. For $K > 1$, $i \neq j \leq k$ and a path $p \in P_u^*(s_i, t_j)$, we define the *clustered guarantee region* as $\Gamma_{V_i V_j}(K,p) = \{v \in V | v \in p \cup V_i \cup V_j \vee \exists\, q \in P(s_i, t_j)\ (v \in q \wedge l(q) < \frac{1}{K}(u(p) + \sigma_i + \tau_j))\}$. We can prove that such a set of nodes has the following approximation property: for $u \in V_i, v \in V_j$ (where $i \neq j$), $p^* \in P^*(u,v)$, $q^* \in P^*[\Gamma_{V_i V_j}(K,p)](u,v)$, we have $c(q^*) \leq K c(p^*)$. Again, clustered guarantee regions may not be the minimially-sized sets having this approximation property, but they are no worse than those generated by any other path.

The definition of guarantee regions imply that the most expensive part of their (pre-processing) computation is finding all paths $p \in P(s,t)$ s.t. $l(p) < H$. There exists in fact a polynomial algorithm that computes all nodes on a path with a total cost $< H$ from a node s to a node t. Computing H itself is straightforward, since it requires knowing $p^* \in P^*_u(s,t)$ and, in the clustered case, upper bounds to the cost of shortest paths in a small set of nodes.

3 Preliminary computational experiments

We used a subgraph of France's road network, roughly corresponding to Île-de-France (i.e. Paris and surroundings), to validate our approach. This subgraph has roughly 300.000 vertices and 800.000 edges. We ran several bidirectional Dijkstra searches [SS05] on the full graph and on guarantee regions to assess the usefulness of our heuristic, with source and destination node chosen at random, and for each source-destination pair we repeated the query 5 times with arc costs generated at random with a uniform distribution each time; upper bounds on arc costs are between 5-10 times lower bounds. We recorded solution quality and CPU times in Table 1. For each value of K (first column), we indicate the average number D of nodes explored in bi-directional Dijkstra searches n the full graph, the average number R of nodes explored in bi-directional Dijkstra searches on the guarantee regions, the average percentage increase P of the approximated solution value with respect to the optimum (0% means that the approximated solution is optimal), the average CPU time savings C in percentage of the CPU times taken by the exact algorithm (0% means as slow as the exact algorithm).

K	D	R	P	C
3	74559	74532	0%	0%
4	74779	74219	0%	0%
5	74651	65126	0%	8.39%
6	74739	39282	0%	46.85%
7	74647	5609	0.07%	93.86%

Table 1
Computational results on unclustered graph: mean values.

To validate the clustered approach, we generated a valid covering of V, and then, for some random cluster pairs, compared the number of explored and settled nodes between a bidirectional Dijkstra search and a bidirectional Dijkstra search constrained to the guarantee regions, where source and destination node of Dijkstra's search where chosen randomly in their respective cluster, performing 5 queries with arc costs generated at random for each source-destination pair. Results are reported in Table 2 (same column labels as Table 1); cluster size was set to 500 nodes.

K	D	R	P	C
6	74493	73262	0%	0%
7	74605	66804	0%	5.83%
8	74129	56761	0%	20.35%
9	74436	34091	0.02%	54.26%
10	74494	13978	1.20%	82.05%

Table 2
Computational results on clustered graph: mean values

4 Conclusion

We have proposed a two-phase Polynomial Time Approximation Scheme for the Point-To-Point Shortest Path Problem, based on a pre-processing phase where we compute guarantee regions where we can restrict the search, and a query phase where we answer point-to-point queries. This was motivated by the need for finding short paths on large-scale graphs very quickly, where arc costs (representing travelling times given by real-time traffic data on a road network) may vary within a given range. The approach was successfully validated on a graph (of limited size) derived from the road network of the Île-de-France region, showing great computational time reduction in query resolution while still finding an optimal or near-optimal solution.

In practice, real-time traffic data is only available for a relatively small subset of all arcs, so that knowing how to infer unknown traffic data from the existing data is also an issue. Although we do not target this particular aspect of the problem in this paper, future work will be undertaken on the matter.

References

[GKW05] A.V. Goldberg, H. Kaplan, and R.F. Werneck. Reach for A^*: Efficient point-to-point shortest path algorithms. Technical Report MSR-TR-2005-132, Microsoft Research, 2005.

[MG04] R. Montemanni and L. M. Gambardella. An exact algorithm for the robust shortest path problem with interval data. *Computers & Operations Research*, 31:1667–1680, 2004.

[MGD04] R. Montemanni, L. M. Gambardella, and A. V. Donati. A branch and bound algorithm for the robust shortest path problem with interval data. *Operations Research Letters*, 32:225–232, 2004.

[SP06] A. Sengupta and T. K. Pal. Solving the shortest path problem with interval arcs. *Fuzzy Optimization and Decision Making*, 5:71–89, 2006.

[SS05] P. Sanders and D. Schultes. Highway hierarchies hasten exact shortest path queries. In G. Stølting Brodal and S. Leonardi, editors, *ESA*, volume 3669 of *Lecture Notes in Computer Science*, pages 568–579. Springer, 2005.