Fast Computation of Shortest Smooth Paths and Uniformly Bounded Stretch.

Margaux Schmied, David Coudert

Context

We study the shortest smooth path problem, which is motivated by traffic-aware routing in road networks. The goal is to compute the fastest route according to the current traffic situation while avoiding undesired detours. For this we have examined the article by Tim Zeitz.
Example:
If we go from Nice to Paris (a long trip) we don't want to leave the highway to gain only 30 sec.

CH-Potentials

Intuitively, vertices on highways are more important than vertices on some rural street.

We contracted vertices successively by ascending importance.
To contract a vertex means to temporarily remove it from the graph while inserting shortcut arcs between more important neighbors to preserve shortest distances among them.

This preprocessing is applied for each algorithm.

Iterative path blocking (IPB)

The algorithm repeats two steps until a valid path is found.

- First step: The shortest path compared to the traffic is calculated by avoiding blocked paths thanks to Dijkstra
- Second step: We check that each sub-path does not violate the UBS constraint. All violating subpaths are added to the blocked paths list
- Third step: We start again until we have a valid path

Iterative path fixing (IPF)

- First step: Find the shortest path with respect to traffic
- Second step: Replace each UBS violating subpath with the shortest path without traffic
- Third step: We continue iteratively replacing the violating segments

Results

ϵ		Increase [%]	Iterations	Blocked paths	Running time [ms] A*	Running time [ms] UBS	Running time [ms] Total	Failed [%]
0.01	IPB-E	0.43	137.90	676.2	307.6	22.7	335.9	2.4
	IPB-H	0.56	22.38	24.9	52.8	21.0	74.0	0.6
	IPF	0.61	1.73	-	-	-	2.3	0.0
0.05	IPB-E	0.34	68.10	351.7	132.5	14.8	150.3	0.9
	IPB-H	0.39	32.78	39.8	19.6	38.7	58.6	0.5
	IPF	0.41	1.54	-	-	-	2.3	0.0
0.10	IPB-E	0.27	47.35	256.4	103.3	12.7	118.3	0.8
	IPB-H	0.33	27.10	27.1	3.5	28.9	32.7	0.3
	IPF	0.34	1.45	-	-	-	2.7	0.0
0.20	IPB-E	0.23	24.92	141.7	51.1	7.5	59.7	0.4
	IPB-H	0.26	19.33	19.0	2.6	19.6	22.4	0.2
	IPF	0.28	1.36	-	-	-	2.1	0.0
0.50	IPB-E	0.16	13.64	80.0	41.1	3.8	45.6	0.1
	IPB-H	0.17	19.54	18.9	2.5	19.4	22.1	0.2
	IPF	0.19	1.26	-	-	-	2.0	0.0
1.00	IPB-E	0.11	10.51	55.5	28.1	4.4	33.4	0.2
	IPB-H	0.12	15.13	14.3	2.4	9.6	12.2	0.1
	IPF	0.14	1.19	-	-	-	2.5	0.0

Average performance of our implementations for different values of ϵ with 1h queries on OSM Europe with synthetic live traffic. The Increase column denotes the length increase with respect to w* of the shortest smooth path over the shortest w∗ path. It includes only values from successful queries.

All other columns indicate average values over all queries, including the ones terminated after 10 seconds.

Smooth Paths

- P a path
- w(P) the weight of a path P
- $D_w(v_1,v_k)$ the shortest distance between nodes v_1 and v_k
- Stretch of a path: $S_w(P) = \frac{w(P)}{D_w(v_1, v_k)}$
- Uniformly bounded stretch: $UBS_w(P) = \max\{0 \le i < j \le k\} S_w(P_{i,j})$

A path P is **ε**-smooth with respect to a weight function w when $UBS_w(P) < 1 + \varepsilon$.

Efficient UBS computation

The idea is to focus on the validity of the path rather than calculating the exact UBS.
We are looking for the shortest path in relation to the traffic from the source with Dijkstra as well as all the distances.
Thus we can check that all the sub-paths are valid in constant time.

Results

	Vertices [$\cdot 10^6$]	Edges [$\cdot 10^6$]	Preprocessing [s] Phase 1	Preprocessing [s] Phase 2
DIMACS Europe	18.0	42.2	2 260.7	11.3
OSM Europe	173.8	348.0	4 270.0	58.8
OSM Germany	11.1	26.2	1 314.0	7.5

Average performance of our implementations of IPB-E, IPB-H and IPF for different query sets on all instances with ϵ = 0.2.

The Increase column denotes the length increase with respect to w∗ of the obtained path over OPTw∗ and includes only successful queries. The running time column also includes the running time of queries aborted after 10 seconds.

		Increase [%] IPB-E	Increase [%] IPB-H	Increase [%] IPF	Running time [ms] IPB-E	Running time [ms] IPB-H	Running time [ms] IPF	Failed [%] IPB-E	Failed [%] IPB-H	Failed [%] IPF
1h	DIMACS Eur Syn	0.8	2.5	4.1	718.0	168.4	1.5	6.4	1.6	0.0
	OSM Eur Syn	0.2	0.3	0.3	59.8	22.4	2.7	0.4	0.2	0.0
	OSM Ger Fri	0.2	1.5	2.2	1 373.7	219.1	4.7	12.7	1.2	0.0
	OSM Ger Tue	0.1	0.3	0.4	261.5	9.2	1.8	2.2	0.0	0.0
4h	DIMACS Eur Syn	0.8	3.4	5.1	3 513.6	435.4	6.1	33.1	3.1	0.0
	OSM Eur Syn	0.2	0.3	0.3	331.4	73.2	8.4	2.1	0.6	0.0
	OSM Ger Fri	0.2	2.1	4.2	6 597.1	2 568.1	89.2	63.4	15.8	0.0
	OSM Ger Tue	0.1	0.4	0.5	1 449.3	93.1	9.8	13.1	0.0	0.0
Random	DIMACS Eur Syn	0.8	2.8	5.3	6 700.6	2 436.7	30.6	64.2	17.1	0.0
	OSM Eur Syn	0.2	0.4	0.4	4 758.3	654.2	140.3	38.9	3.1	0.0
	OSM Ger Fri	0.2	2.1	4.1	5 771.1	2 419.8	84.5	56.0	16.3	0.0
	OSM Ger Tue	0.1	0.4	0.5	1 366.0	111.6	9.6	12.0	0.0	0.0

Summary

The author of the article adapts the existing IPB-H algorithm. It outperforms the original implementation by two orders of magnitude. Moreover, it presents the necessary modifications to make the algorithm exact. IPB-E is still about an order of magnitude faster than the CRP-based heuristic implementation. As IPB-H and IPB-E are not always able to find solutions in reasonable time, it introduces another heuristic, IPF. It can still find smooth paths even for random queries on massive continent-sized instances in tenths of milliseconds.

This work has been supported by the French government, through the UCA DS4H Investments in the Future project managed by the National Research Agency (ANR) with the reference number ANR-17-EURE-0004.

* Zeitz, Tim. "Fast Computation of Shortest Smooth Paths and Uniformly Bounded Stretch with Lazy RPHAST." *20th International Symposium on Experimental Algorithms (SEA 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2022.