

```

38         const searchspacewithbuckets &searchspace_with_buckets,
39         std::shared_ptr<std::vector<EdgeWeight>> result_table) const
40     {
41         const NodeID node = query_heap.DeleteMin();
42         const int source_distance = query_heap.GetKey(node);
43
44         // check if each encountered node has an entry
45         const auto bucket_iterator = search_space_with_buckets.find(node);
46         // iterate bucket if there exists one
47         if (bucket_iterator != search_space_with_buckets.end())
48         {
49             const std::vector<NodeBucket> &bucket_list = bucket_iterator->second;
50             for (const NodeBucket &current_bucket : bucket_list)
51             {
52                 // get target id from bucket entry
53                 const unsigned target_id = current_bucket.target_id;
54                 const int target_distance = current_bucket.distance;
55                 const EdgeWeight current_distance =
56                     (*result_table)[source_id * number_of_locations + target_id];
57                 // check if new distance is better
58                 const EdgeWeight new_distance = source_distance + target_distance;
59                 if (new_distance > 0 && new_distance < current_distance)
60                 {
61                     (*result_table)[source_id * number_of_locations + target_id] =
62                         (source_distance + target_distance);
63                 }
64             }
65         }
66         if (StallAtNode<true>(node, source_distance, query_heap))

```

Everything But Directions

All the Other Exciting Things You Can Do With Routing

Dennis Luxen

Mapbox / Project OSRM

dennis@mapbox.com

Today's Agenda

1. Overview of (Route) Labeling Algorithms
2. Applications
 - Closest Dispatch
 - Walkability
 - Distance Tables
3. Vandalism Detection



(Route) Labeling Algorithms

Intuitive Approach to Route Planning

Observation 1:

Almost all long-distance routes enter the arterial network at one point.

Observation 2:

The routes to almost everywhere lead over a few but very important locations in the road network.

Intuitive Approach to Route Planning

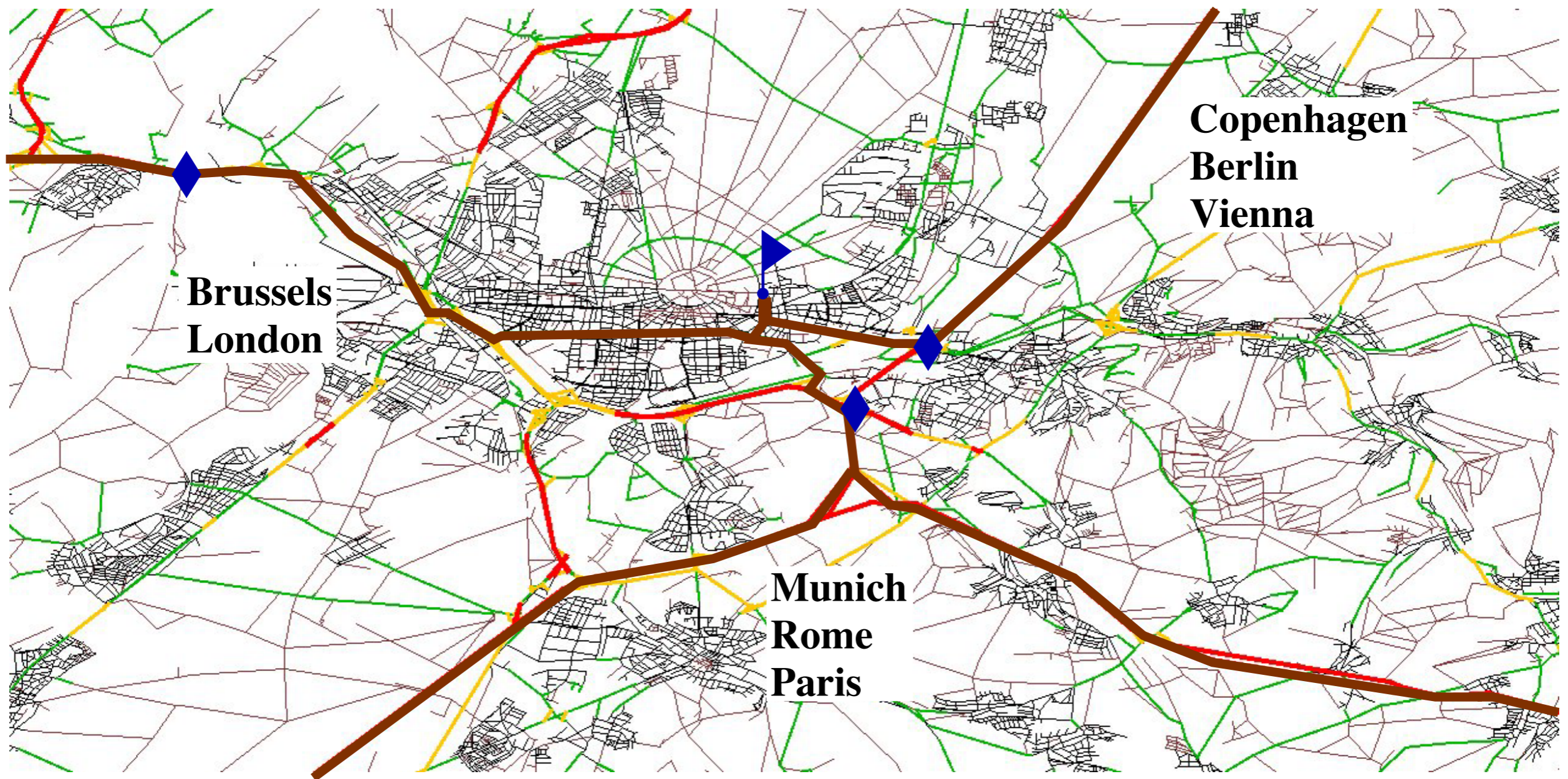
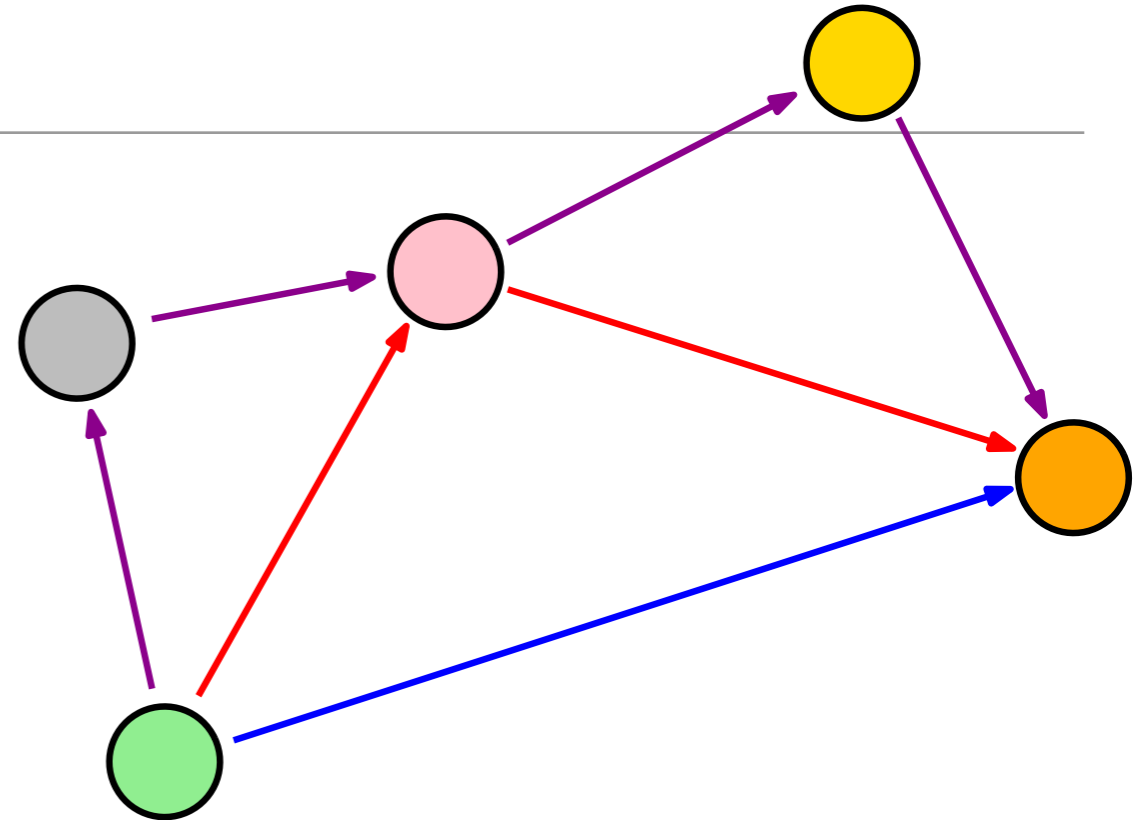


Image from a previous life in academia: <http://algo2.iti.kit.edu>

(Route) Labeling Algorithms*

Our current approach:

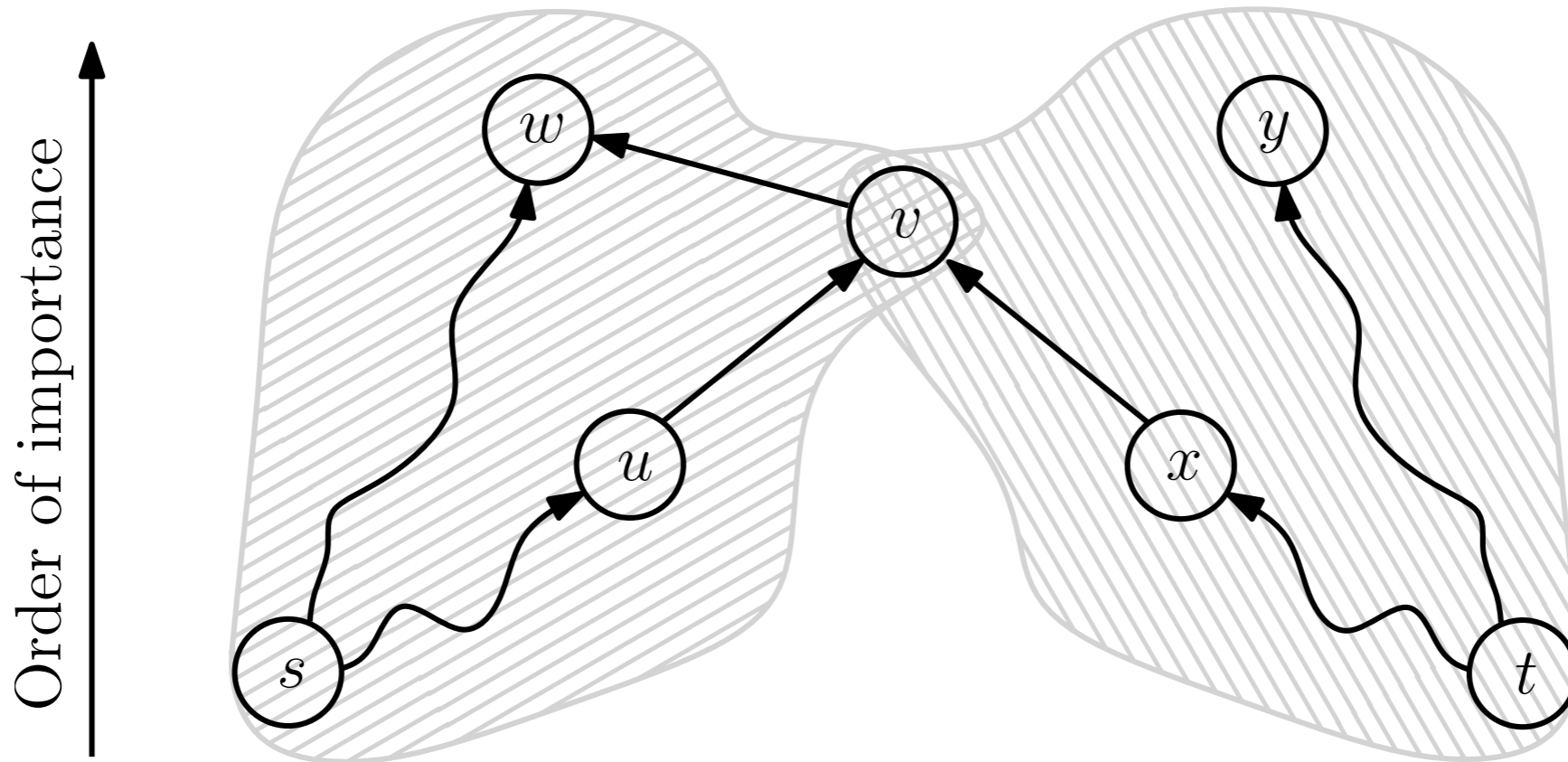
1. Preprocessing the input data:
 - discover layers of *importance*
 - add additional *structure*
2. Leverage this data for fast queries
 - ignore most of the data
 - layaway from one piece of structure to the next
 - search spaces are small



*
omitting all the details

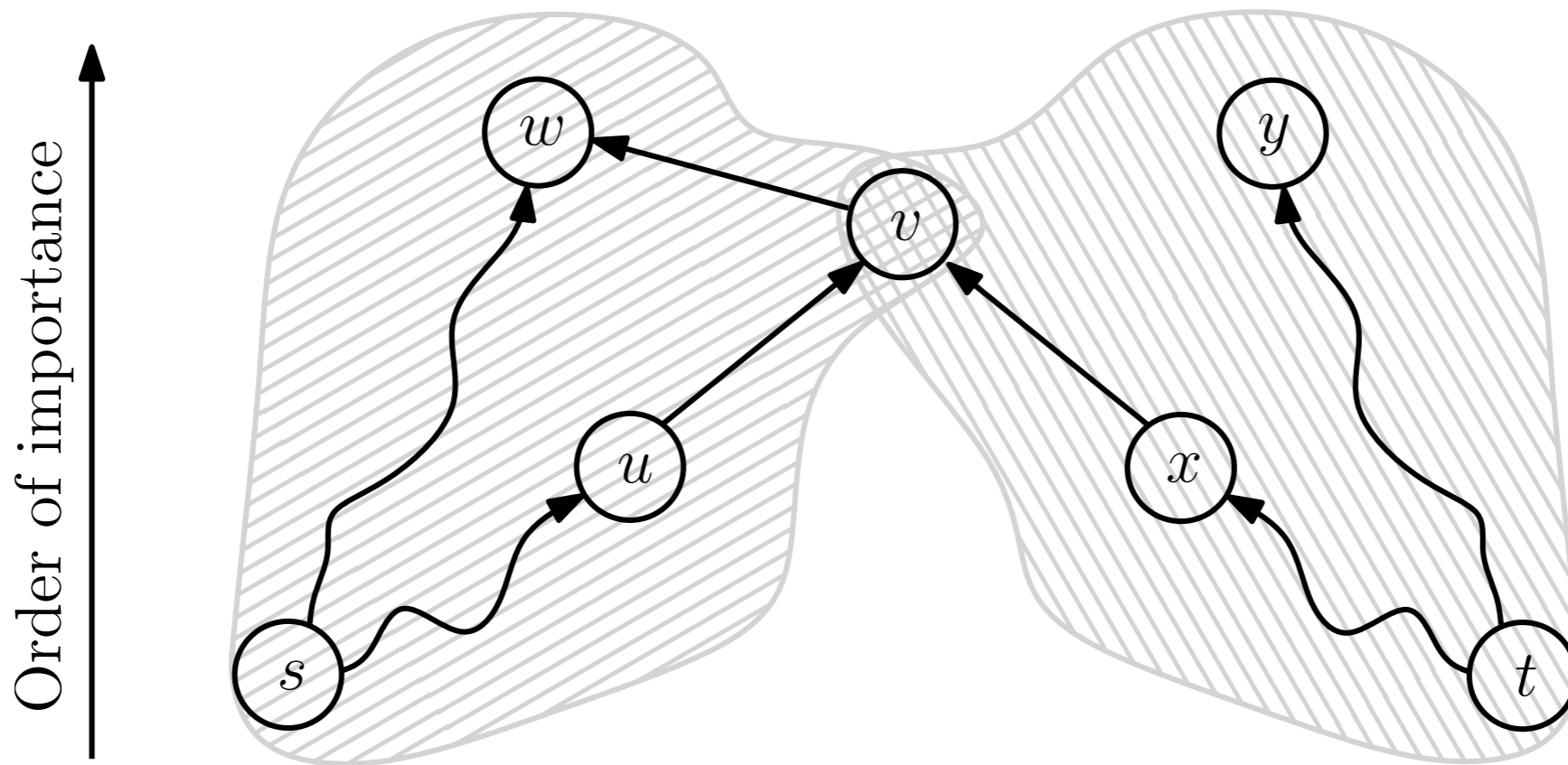
(Route) Labeling Algorithms

Searching simultaneously from source and target location guarantees the **labeling** of one piece of road with the minimum *distances* from both ends.



(Route) Labeling Algorithms

Finding a route is now formulated as looking at the intersection of two sets.



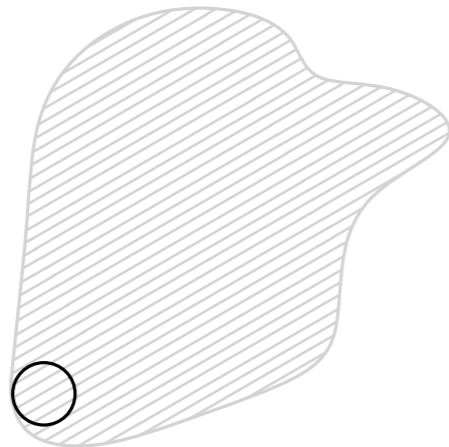
Also note: search spaces of a location always the same

In a nutshell

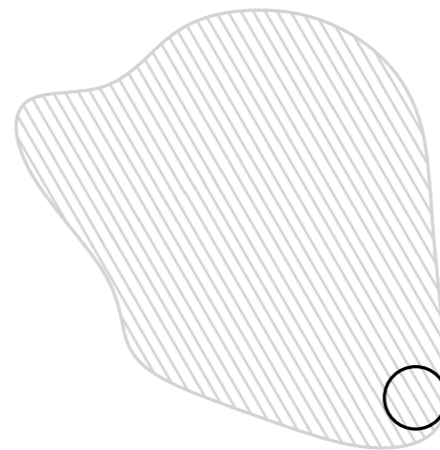
1. Search space are small subsets of data:
 - forward search from source
 - reverse search from target
2. Paths are found by intersecting both
 - Fast search: Milliseconds even for long-distance routes

Notation:

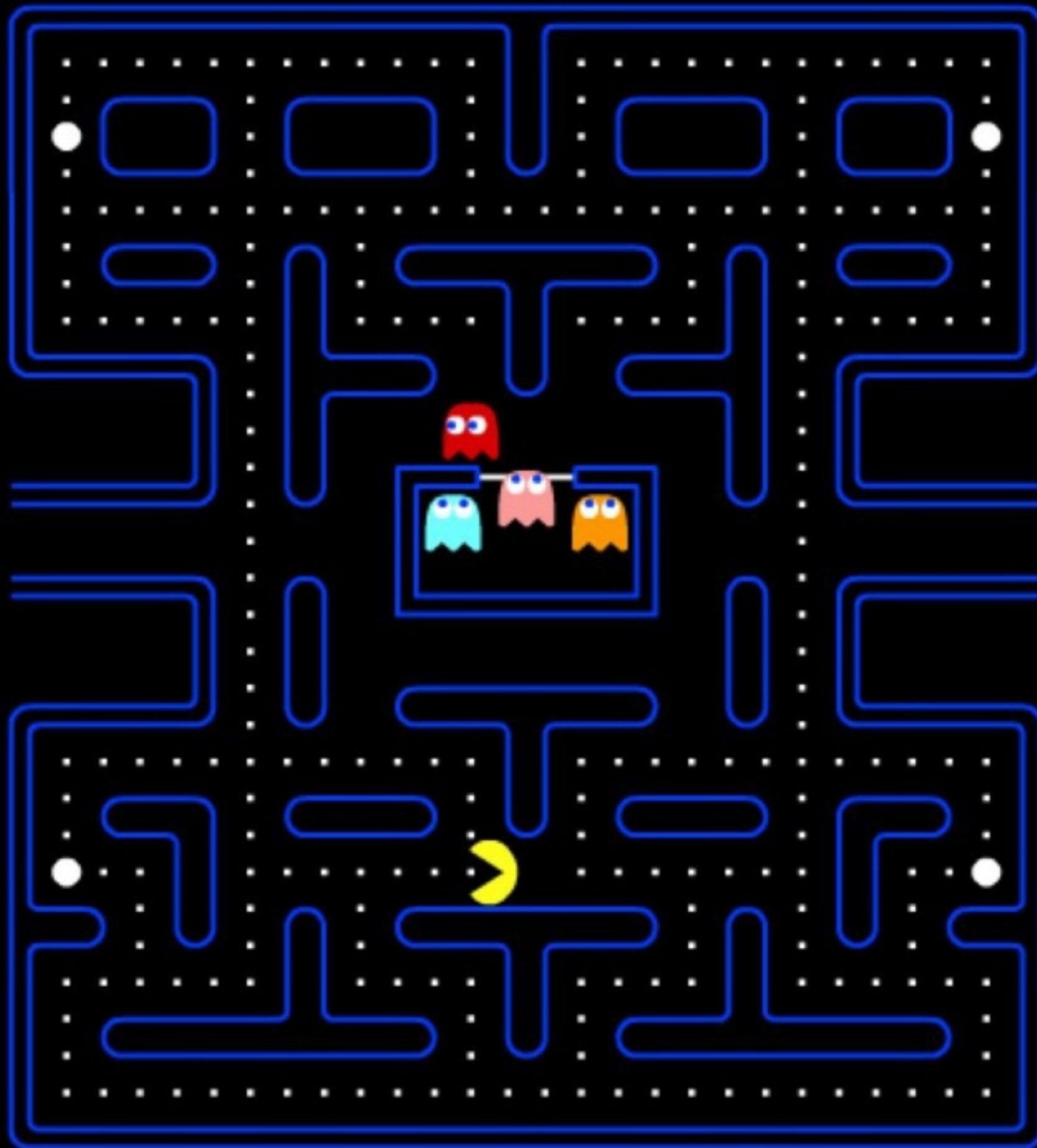
forward search



reverse search



Application: Closest Dispatch



SCORE 

LIVES 



Closest Dispatch

Setting:

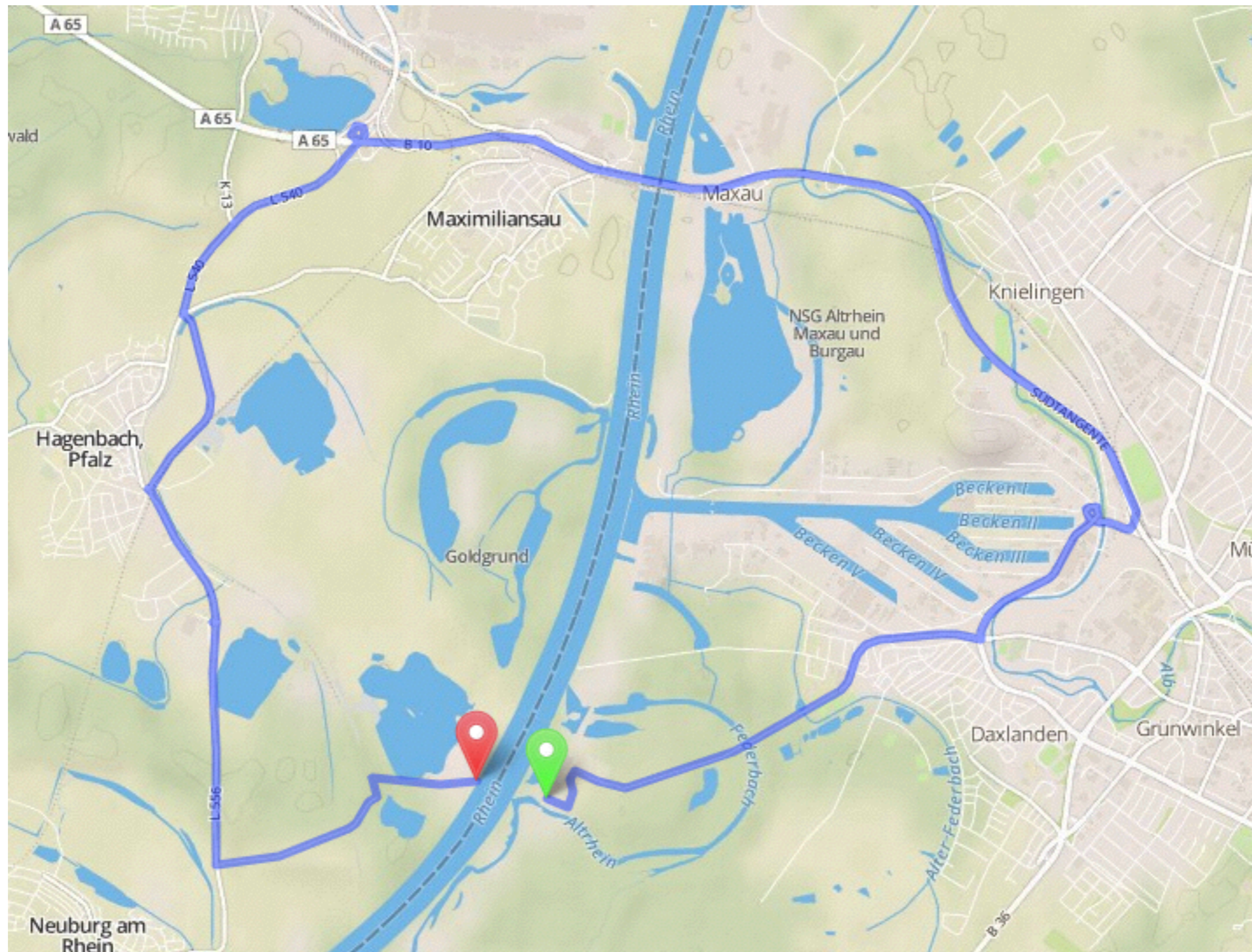
- multiple vehicles on patrol
- for a given location find the nearest one to respond

Examples:

- Taxis looking for customers
- Police cars/bikes/peds on patrol

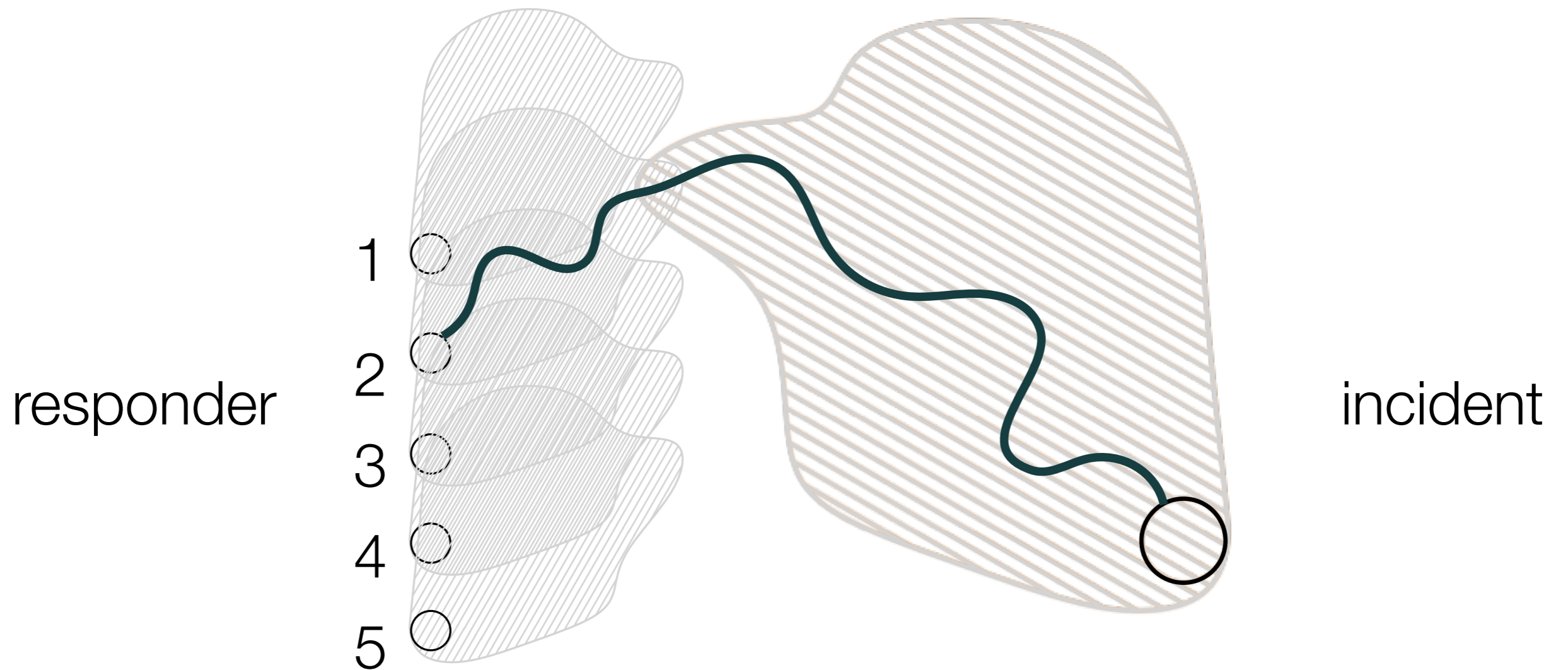
Essentially, this is a many-to-one path problem!

Closest Dispatch - Network vs Crows Fly Distance



Closest Dispatch - Modeling

UPDATE stored *reverse* search spaces w/ GPS ping



Intersect *incident* with the preprocessed data for best responder

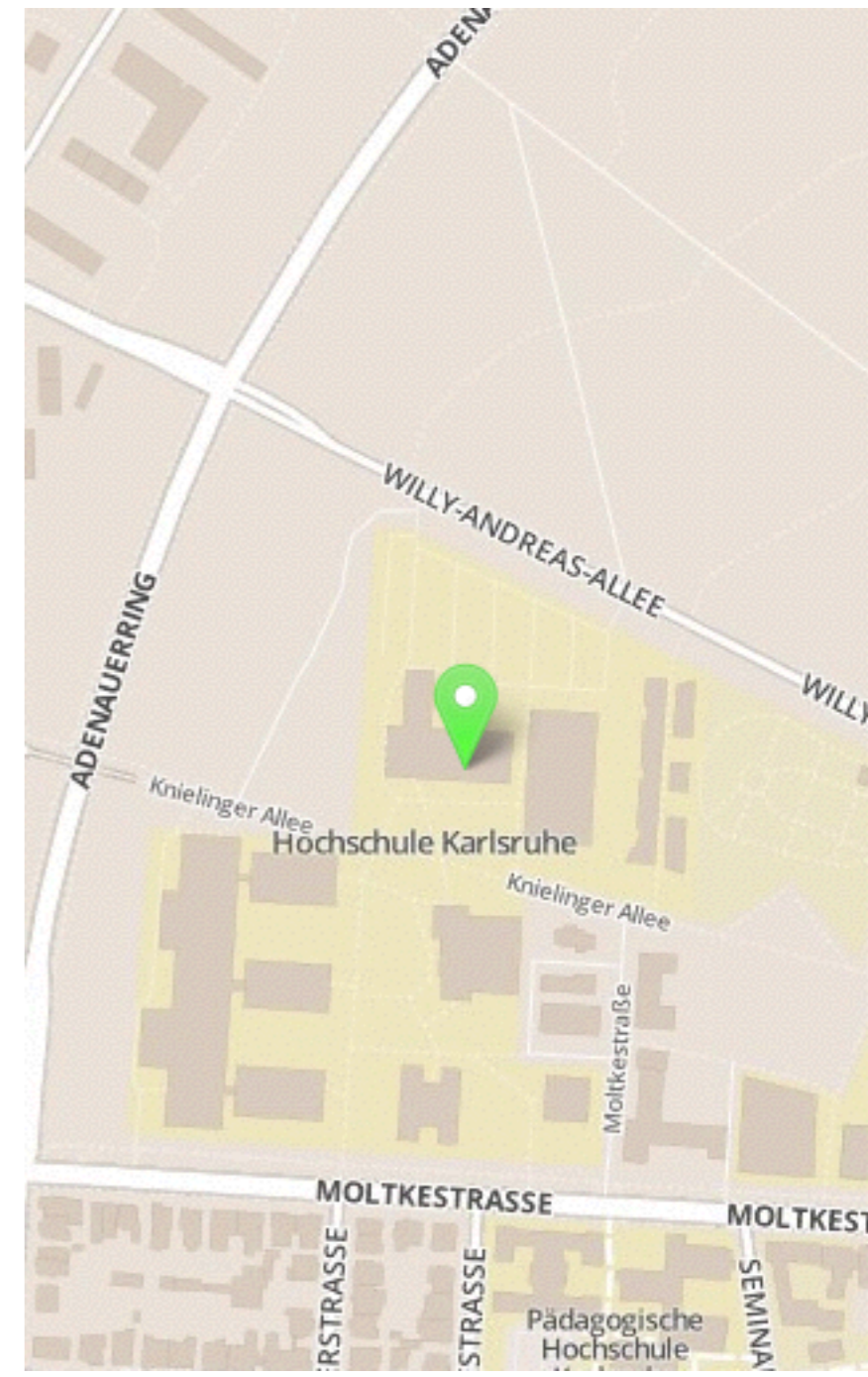
Application: Walkability

How walkable is a location?

We need to look at a location and determine distance to a nearest:

- grocery store
- doctors office
- bar
- italian/chinese/mexican/... restaurant
- bus or tram stop
-

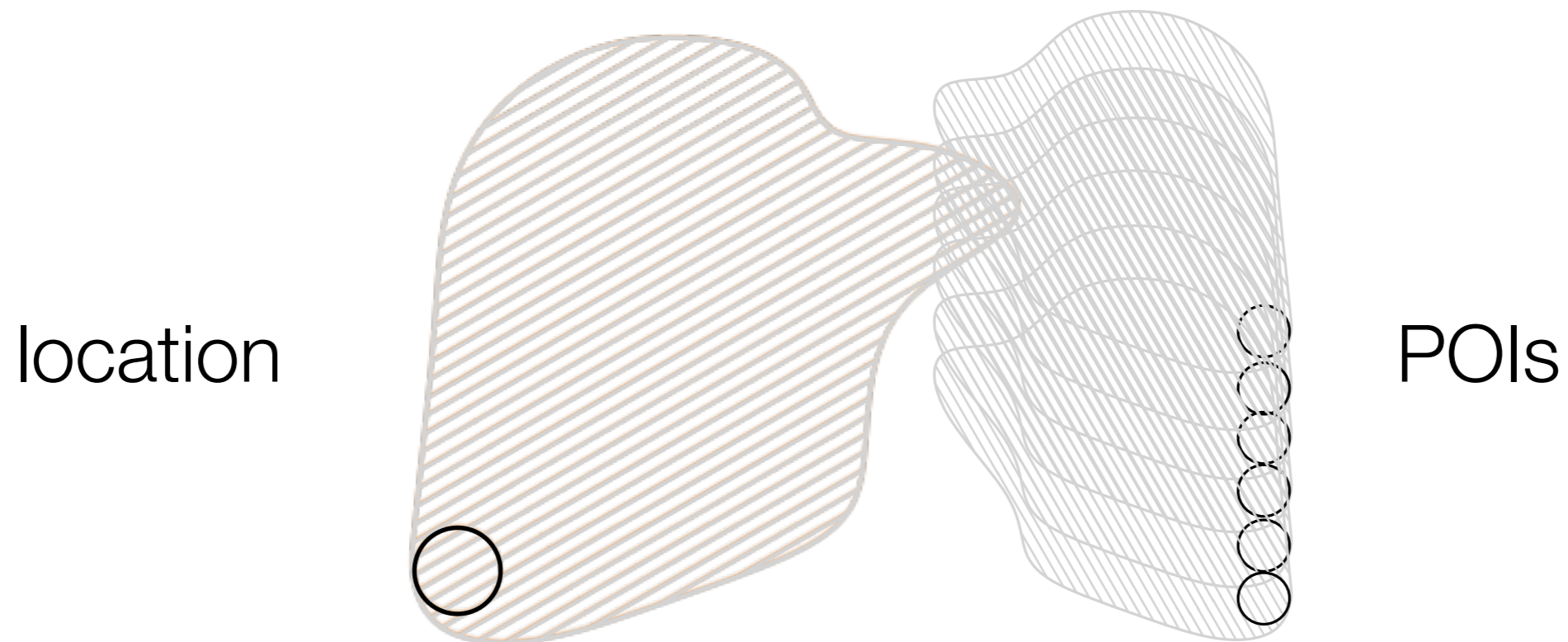
Essentially this is a one-to-many problem!



Walkability - Modeling

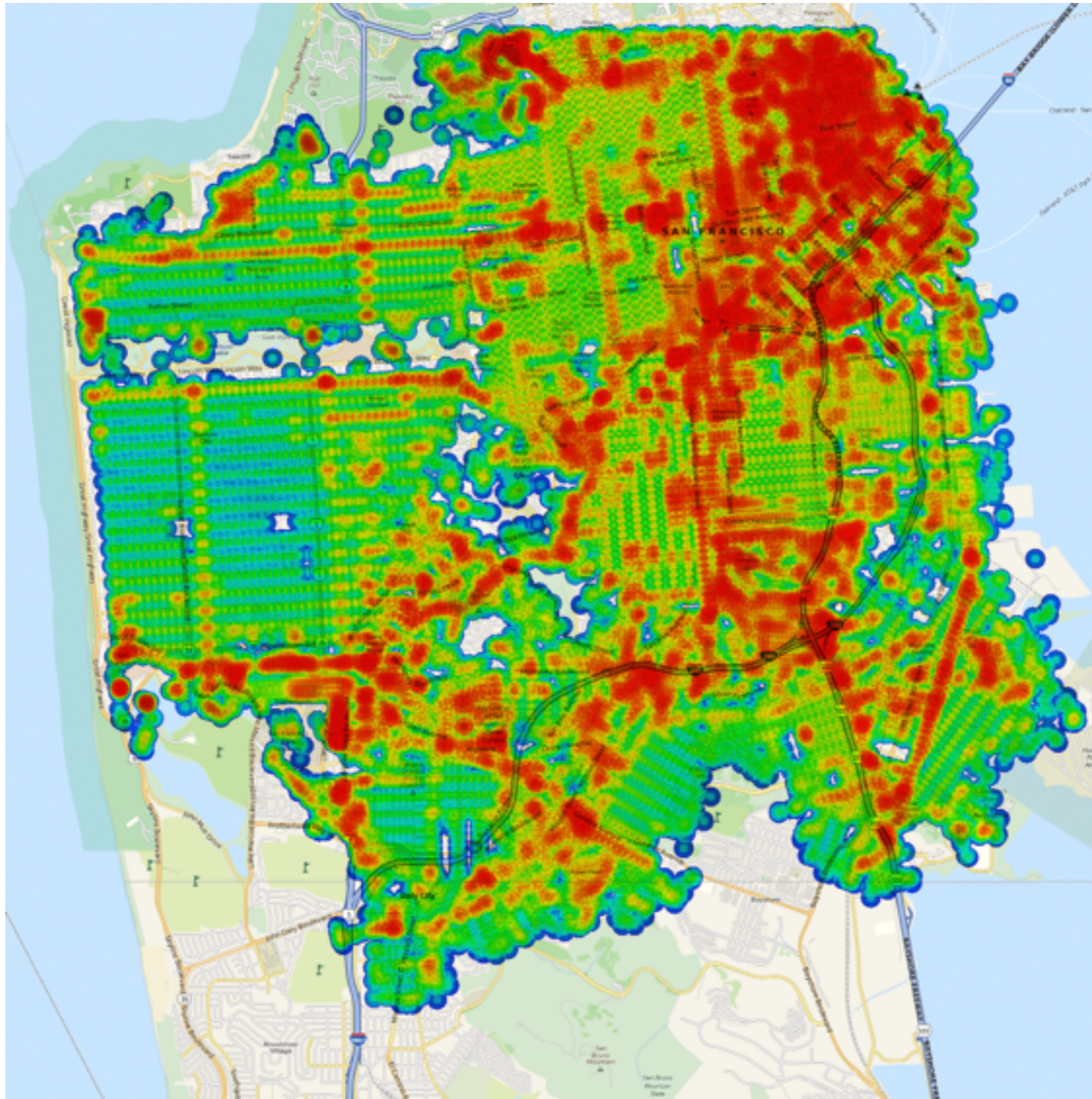
- Many fixed locations -> targets
- One query location -> one source

Preprocessing simple: reverse search space for each POI



Query: Get one forward search space and aggregate distances

Walkability - Results for San Francisco



Application: Distance Tables

Distance Tables*

Given sets of source and target locations, compute all pair-wise distances.

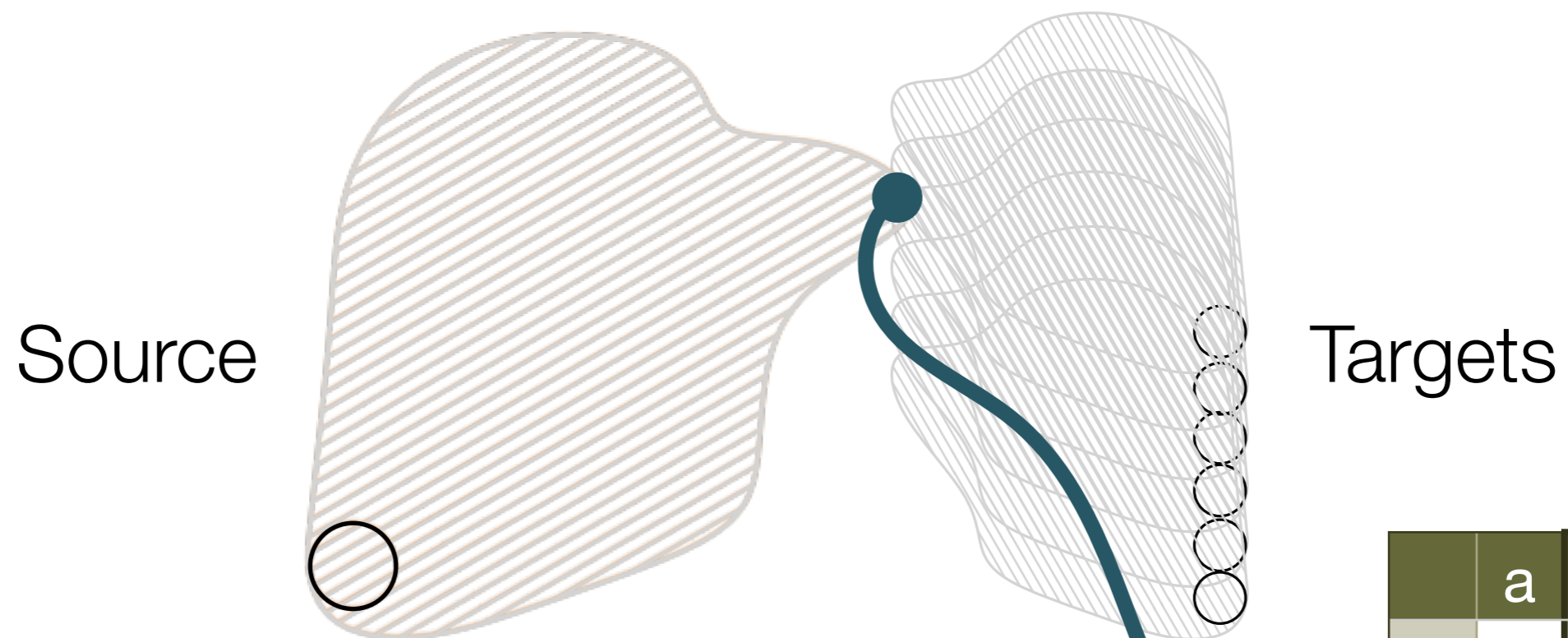
674	674	569	493	464	218	850	566	823	709	212	861	566	581	358	692	248	137	691	931	662	469	833	642	1071	761	643	1131	716	311	1133	1098	80
569	454	454	378	422	1003	1250	159	1223	1109	612	1261	108	981	333	579	648	537	17	1331	306	869	1233	1042	1471	1161	1043	1531	1116	711	1658	1498	594
493	378	134		29	822	1069	209	923	928	431	1080	373	800	157	259	257	356	395	1150	366	688	1052	861	1290	980	862	1350	935	530	1477	1317	413
464	422	105	29		793	1040	248	903	899	402	1051	314	771	106	230	286	327	366	1121	395	659	1023	832	1261	951	833	1321	906	501	1448	1288	384
218	1003	898	822	793		657	844	630	516	391	668	895	388	687	1021	498	466	1020	738	991	276	640	449	878	568	450	938	523	292	1065	905	296
850	1250	1145	1069	1040	657		1091	645	275	657	425	1142	403	934	1268	824	713	1267	167	1238	381	397	342	307	87	207	367	280	539	494	248	770
566	159	295	209	248	844	1091		1064	950	504	1102	51	822	157	491	489	378	176	1172	147	710	1074	833	1312	1002	884	1372	957	552	1499	1390	435
823	1223	798	923	903	630	645	1064		321	611	345	1115	242	907	923	549	686	1240	677	1211	395	579	254	817	507	389	926	462	512	1143	893	743
709	1109	1004	928	899	516	275	950	321		497	286	1001	128	749	1015	683	572	1126	356	1097	240	258	67	496	186	68	556	141	398	683	523	629
212	612	507	431	402	391	657	504	611	497		578	504	369	296	630	119	75	629	719	600	257	621	430	859	549	431	919	504	99	959	799	132
861	1261	1156	1080	1051	668	425	1102	345	286	578		1153	414	945	1279	835	724	1278	506	1249	392	262	353	646	336	218	680	145	550	272	645	781
566	108	346	373	314	895	1142	51	1115	1001	504	1153		873	208	471	540	429	125	1223	147	761	1125	934	1363	1053	935	1423	1008	603	1550	1390	486
581	981	876	800	771	388	403	822	242	128	369	414	873		665	999	555	444	998	484	969	112	386	61	624	404	197	616	269	270	686	741	501
358	333	211	157	106	687	934	157	907	749	296	945	208	665		334	332	221	333	1015	304	553	917	726	1155	845	727	1215	800	395	1342	1182	278
692	579	125	259	230	1021	1268	491	923	1015	630	1279	471	999	334		374	555	596	1349	609	889	1253	1060	1489	1179	1061	1549	1134	729	1676	1516	612
248	648	249	257	286	498	824	489	549	683	119	835	540	555	332	374		111	665	905	636	443	807	616	1045	735	617	1105	690	285	1232	1072	168
137	537	432	356	327	466	713	378	686	572	75	724	429	444	221	555	111		554	794	525	332	696	505	934	624	506	994	174	1121	961	57	
691	17	472	395	366	1020	1267	176	1240	1126	629	1278	125	998	333	596	665	554		1348	272	886	1250	1059	1488	1178	1060	1548	1133	728	1675	1515	611
931	1331	1226	1150	1121	738	167	1172	677	356	719	506	1223	484	1015	1349	905	794	1348		1319	487	123	448	210	170	288	236	361	620	469	415	851
662	306	442	366	395	991	1238	147	1211	1097	600	1249	147	969	304	609	636	525	272	1319		857	1221	1030	1459	1149	1031	1519	1104	699	1646	1486	582
469	869	764	688	659	276	381	710	395	240	257	392	761	112	553	889	443	332	886	487	857		364	173	602	292	174	662	247	158	789	629	389
833	1233	1128	1052	1023	640	397	1074	579	258	621	262	1125	386	917	1253	807	696	1250	123	1221	364		325	333	308	190	652	117	522	534	617	753
642	1042	937	861	832	449	342	833	254	67	430	353	934	61	726	1060	616	505	1059	448	1030	173	325		563	253	135	623	208	331	750	590	562
1071	1471	1366	1290	1261	878	307	1312	817	496	859	646	1363	624	1155	1489	1045	934	1488	210	1459	602	333	563		310	428	60	501	760	187	555	991
761	1161	1174	980	951	568	87	1002	507	186	549	336	1053	404	845	1179	735	624	1178	170	1149	292	308	253	310		118	370	191	450	497	335	681
643	1043	938	862	833	450	207	884	389	68	431	218	935	197	727	1061	617	506	1060	288	1031	174	190	135	428	118		488	73	332	615	455	563
1131	1531	1426	1350	1321	938	367	1372	926	556	919	680	1423	616	1215	1549	1105	994	1548	236	1519	662	652	623	60	370	488		561	820	247	615	1051
716	1116	1011	935	906	523	280	957	462	141	504	145	1008	269	800	1134	690	579	1133	361	1104	247	117	208	501	191	73	561		405	688	500	636
311	711	606	530	501	292	539	552	512	398	99	550	603	270	395	729	285	174	728	620	699	158	522	331	760	450	332	820	405		876	787	231
1133	1658	1553	1477	1448	1065	494	1499	1143	683	959	272	1550	686	1342	1676	1232	1121	1675	469	1646	789	534	750	187	497	615	247	688	876		742	1178
1098	1498	1393	1317	1288	905	248	1390	893	523	799	645	1390	741	1182	1516	1072	961	1515	415	1486	629	617	590	555	335	455	615	500	787	742		1018
80	594	489	413	384	296	770	435	743	629	132	781	486	501	278	612	168	57	611	851	582	389	753	562	991	681	563	1051	636	231	1178	1018	
756	1156	1051	975	946	563	578	997	67	303	544	412	1048	175	840	731	482	619	1173	659	1144	287	561	236	799	489	371	859	444	445	684	826	676
824	1224	1119	1043	1014	631	60	1065	570	249	525	399	1116	377	907	1242	798	687	1241	107	1241	355	345	316	247	63	181	307	228	513	434	308	657
773	1173	1068	993	963	580	673	1014	133	188	561	333	1066	183	887	1181	747	676	1180	554	1161	304	456	131	684	384	366	754	238	463	684	731	682

Essentially, this is a many-to-many path problem!

* <http://router.project-osrm/table?loc=lat,lon&loc=.....&loc=lat,lon>

Distance Tables - Approach

1. Preprocess reverse searches for targets
2. Intersect with forward search for each source



Search complexity: linear instead of quadratic !

	a	b	c	d
a	-	8	6	4
b	23	-	64	4
c	3		-	7
d	35	85	4	-

Vandalism Detection



Vandalism Detection

Goal: Detect data issue automatically by looking at change

Here: Detect vandalism in the road network

Remember?

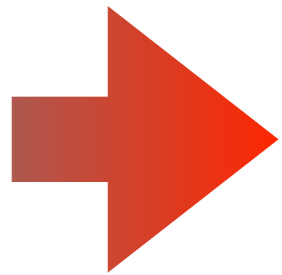
The routes to almost everywhere lead over a few but very important locations in the road network.

Removing an important street will have an impact on many routes, even between randomly selected locations.

Let's cover those roads and look for significant change!

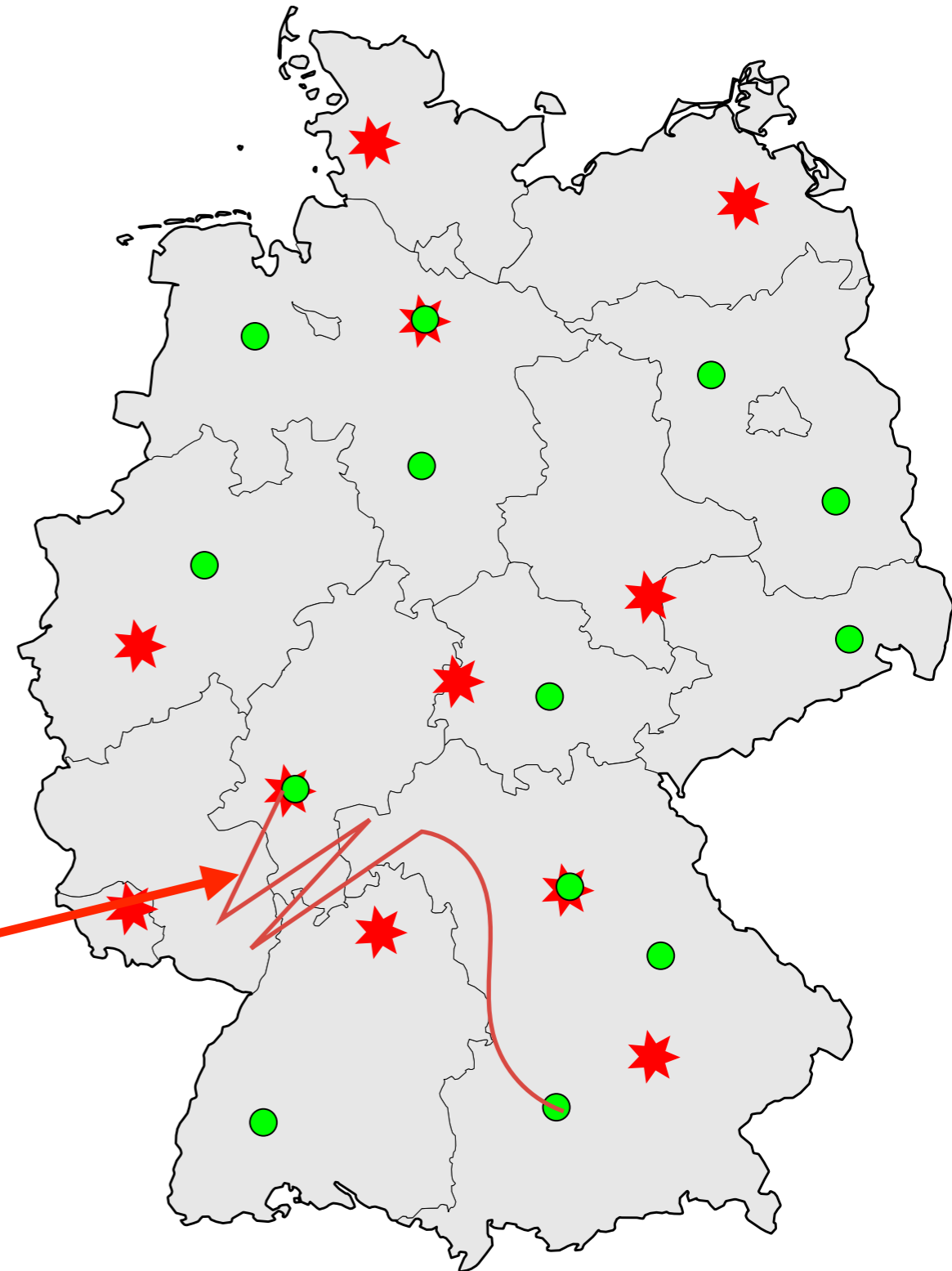
Vandalism Detection

1. Select good sets of locations
2. Compute distance tables
3. Do driving times change?



Look into changes

	a	b	c	d
a	-	8	6	4
b	23	-	64	4
c	3	7	-	7
d	35	85	4	-



Questions?

Email: dennis@mapbox.com

Twitter: @ProjectOSRM

Website: <http://osrm.at>

github: DennisOSRM/Project-OSRM



Yet Another Thing

Ctrl-Alt-Del: Directions from A:\ to Z:\



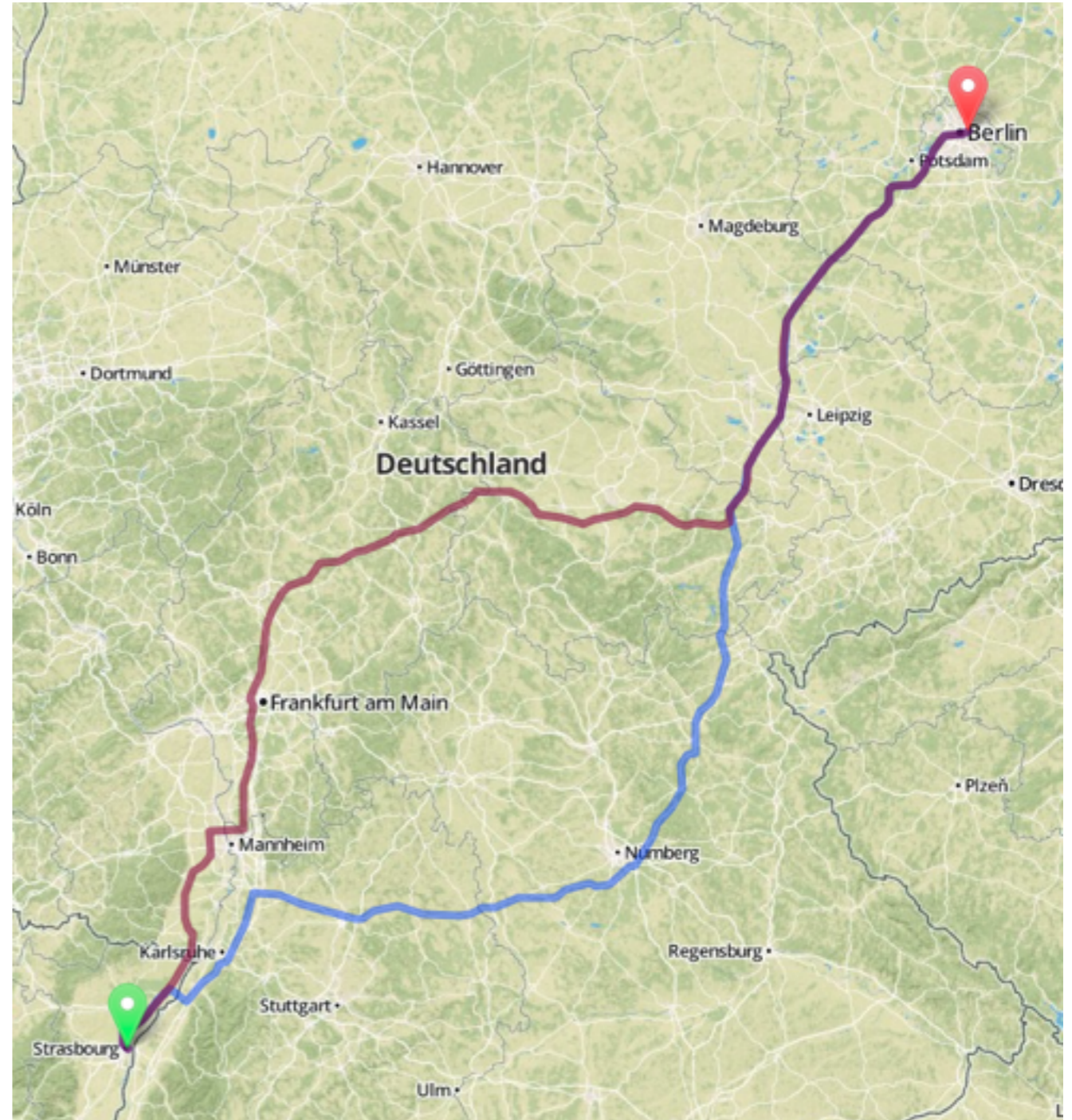
URL: <http://build.project-osrm.org>

Application: Ride Sharing

Ride Sharing

online ride sharing is dull

only city-to-city?

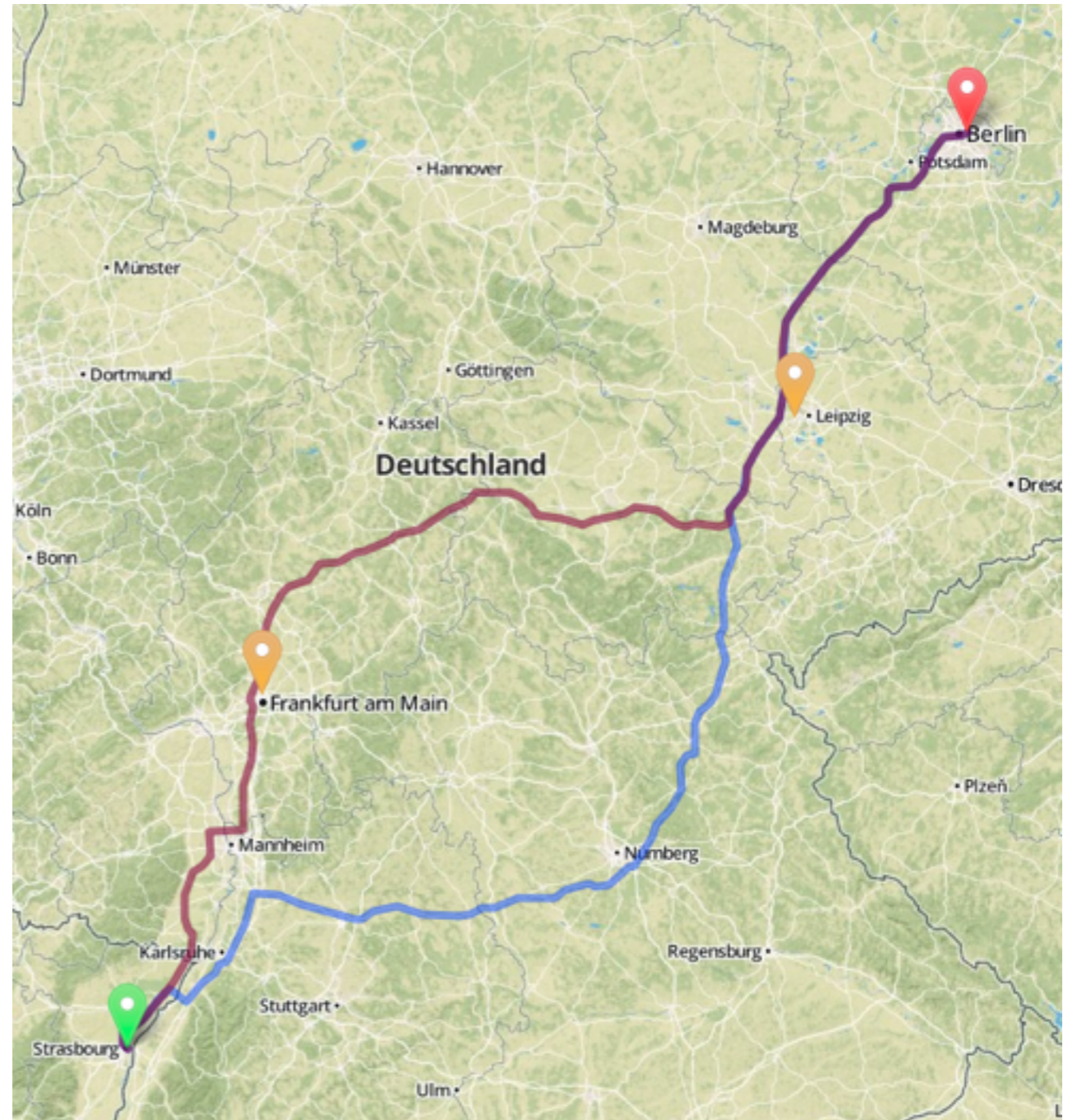


Ride Sharing

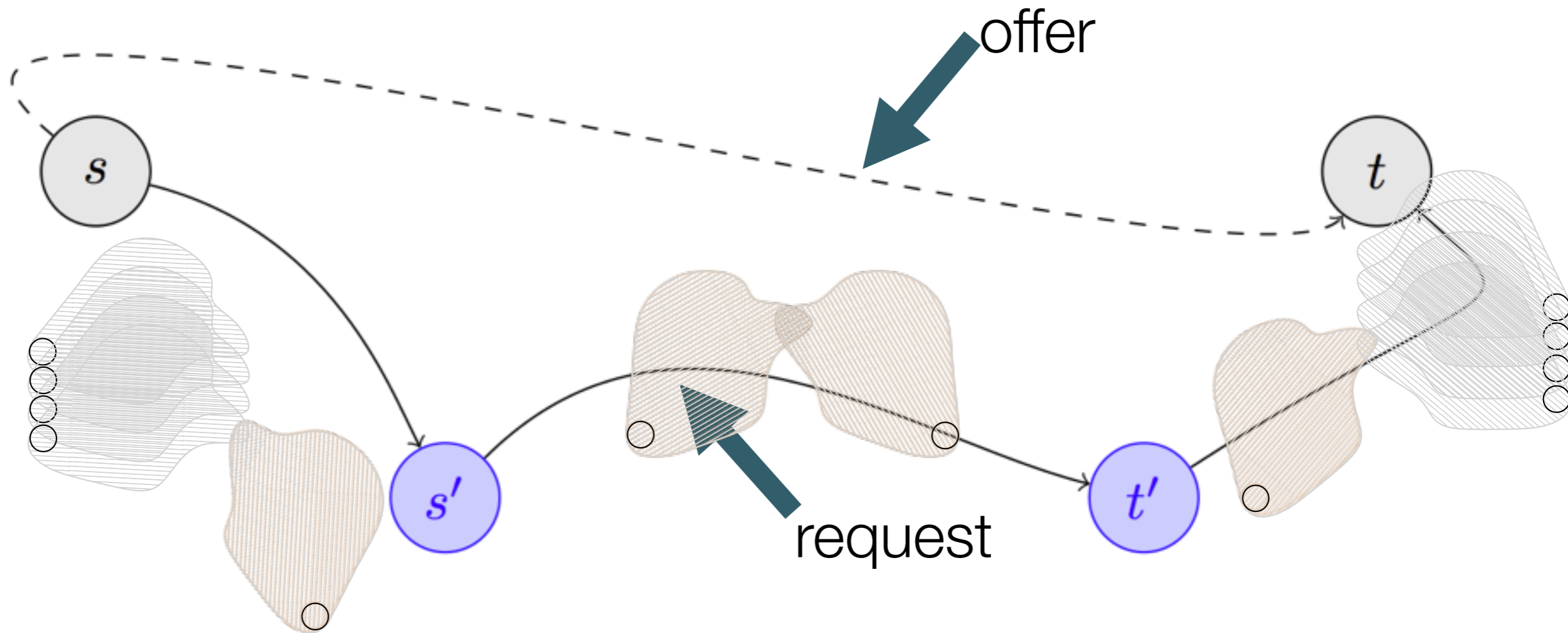
online ride sharing is dull

only city-to-city?

reasonable detours



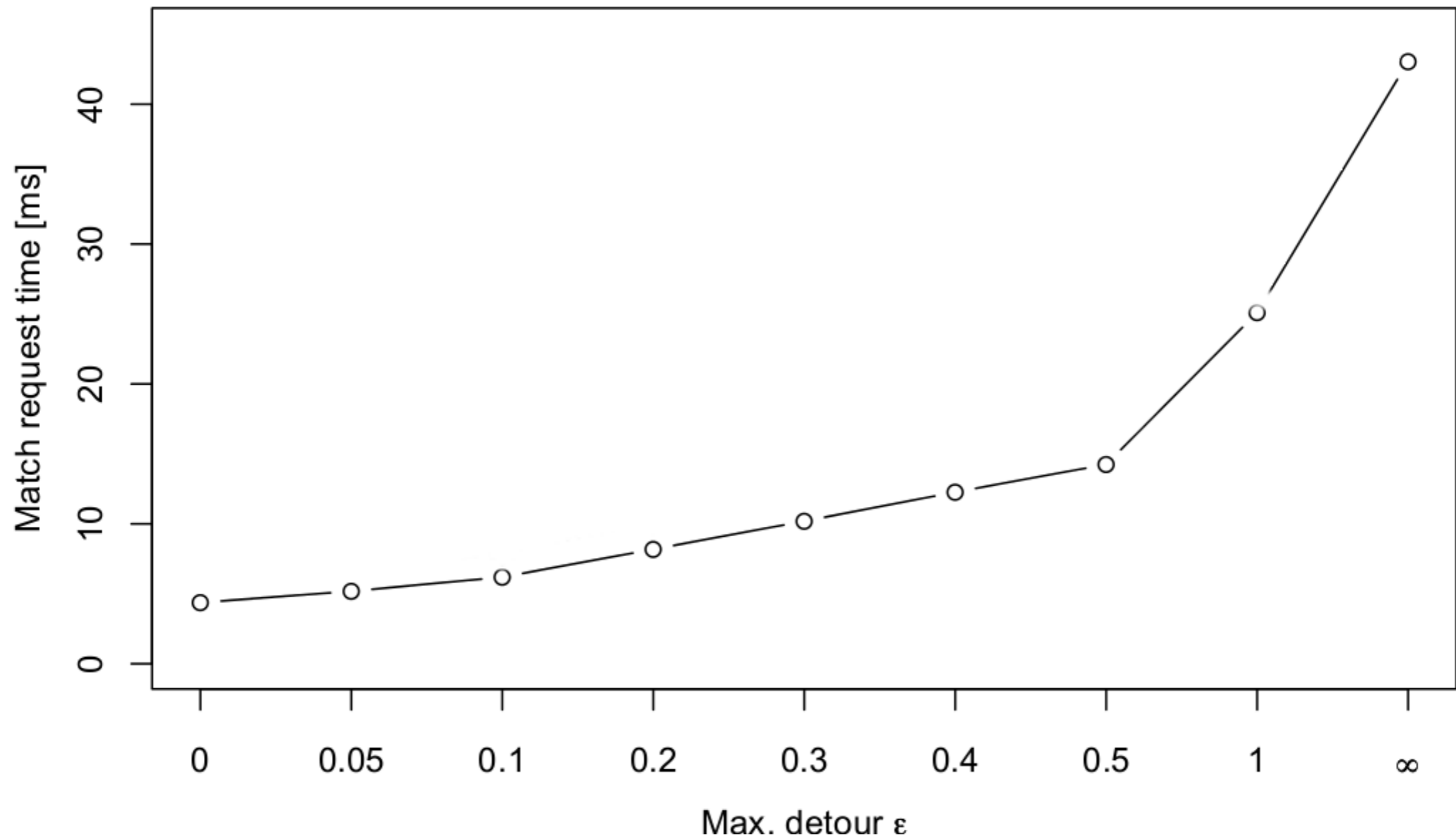
Ride Sharing - modeling



- search spaces for all offers
- matching on the fly

Ride Sharing - Performance

Matching against a set of 100,000 offers





„© OpenStreetMap Contributors“