Fast Routing in Road Networks with Transit Nodes

Article *in* Science · May 2007

Impact Factor: 33.61 · DOI: 10.1126/science.1137521 · Source: PubMed

4 authors, including:

Hannah Bast
University of Freiburg
73 PUBLICATIONS **921** CITATIONS

SEE PROFILE

Fast Routing in Road Networks with Transit Nodes

Holger Bast,[1] Stefan Funke,[1] Peter Sanders,[2]* Dominik Schultes[2]

Computing the quickest route in a road network from a given source to a given target is an important problem used in diverse applications, such as car navigation systems, Internet route planners, traffic simulation, or logistics optimization. Dijkstra's classical algorithm for this problem (*1*) iteratively visits all nodes that are closer to the source than the target before reaching the target. On road networks for a subcontinent like Western Europe or the United States, this takes about 10 s on a workstation, which is too slow for many applications. Commercial systems instead use heuristics that are much faster but that do not guarantee optimal routes. There has been considerable interest in faster techniques for computing exact routes. The Supporting Online Material (SOM) text summarizes previous techniques. These use special properties of road networks and often precompute some connections.

Our approach, transit node routing, has its basis in a simple observation intuitively used by humans: When you drive to somewhere far away, you will leave your current location via one of only a few access routes to a relatively small set of transit nodes interconnected by a sparse network relevant for long-distance travel. We therefore precomputed the following connections: from each potential source or target to its access transit nodes and between all pairs of transit nodes. We also needed an effective notion of "far away": A locality filter determines when the precomputed information guarantees optimal results. Figure 1 gives an example.

For an efficient implementation, we adapted highway hierarchies, the fastest previous approach (*2*). A highway hierarchy consists of a sequence of levels, where level $i + 1$ is constructed from level i by bypassing low-degree nodes and removing edges that never appear far away from the source or target of a quickest path. Interestingly, these levels are geometrically decreasing in size and otherwise similar to each other. Connection queries can be computed efficiently because, sufficiently far away from source and target, only high levels of the hierarchy need to be considered. The nodes of the highest level become the transit node set. For a commercial directed road network for Western Europe (results for North America are similar) with 18,010,173 nodes, we got 11,293 transit nodes, an average of 9.9 access transit nodes, and a locality filter allowing us to treat 98.7% of all queries by using a few table lookups (*3*).

We were not done yet, because the remaining 1.3% of local queries are also important in applications. Therefore, we introduced an additional layer of secondary transit nodes. Because the secondary distance table only needs to store regional connections not covered by the primary layer, memory consumption remains reasonable, although in our example graph we have 323,356 secondary transit nodes. After applying a third layer with 2,954,721 tertiary transit nodes, the remaining local searches involve only a handful of nodes. We obtained query times between 5 µs for global queries and 20 µs for local queries (fig. S1). The precomputation needs less than 3 hours overall, and the precomputed information consumes 4.5 gigabytes. We also have versions with considerably lower space consumption at the cost of more expensive queries.

[1]Max-Planck-Institut für Informatik, 66123 Saarbrücken, Germany. [2]Universität Karlsruhe (TH), 76128 Karlsruhe, Germany.

*To whom correspondence should be addressed. E-mail: sanders@ira.uka.de

References and Notes

1. E. W. Dijkstra, *Numer. Math.* **1**, 269 (1959).
2. P. Sanders, D. Schultes, in *Algorithms: ESA 2006*, 14th European Symposium on Algorithms (ESA), Zurich, Switzerland, 11 to 13 September 2006 (vol. 4168 of *Lecture Notes in Computer Science*, Springer, Berlin, 2006), pp. 804–816.
3. More details can be found at http://algo2.iti.uka.de/schultes/hwy/.
4. This work was partially supported by Deutsche Forschungsgemeinschaft grant SA 933/1-3.

Supporting Online Material

www.sciencemag.org/cgi/content/full/316/5824/566/DC1
SOM Text
Fig. S1
References

13 November 2006; accepted 12 January 2007
10.1126/science.1137521

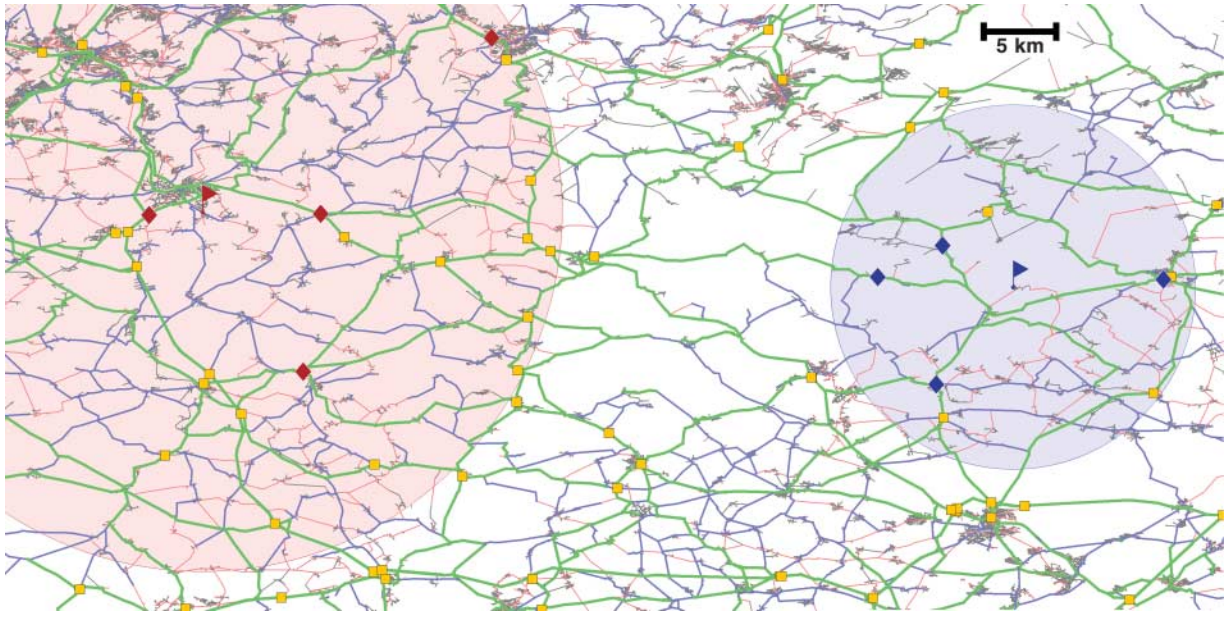

Fig. 1. Finding the quickest route between two points somewhere between Saarbrücken and Karlsruhe (triangle-shaped flags), Germany, amounts to retrieving the 2-by-4 access transit nodes (diamonds), performing 16 table lookups between all pairs of these nodes, and checking that the two disks defining the locality filter do not overlap. The levels of the highway hierarchy are drawn by using the colors gray, red, blue, and green for levels 0-1, 2, 3, and 4, respectively. The other transit nodes are drawn as small yellow squares.