Efficient Profile Routing for Electric Vehicles

René Schönfelder, Martin Leucker, and Sebastian Walther

Institute for Software Engineering and Programming Languages,
University of Lübeck, Germany
{schoenfr,leucker}@isp.uni-luebeck.de, sebastian.walther@web.de

Abstract. This paper introduces a powerful, efficient and generic framework for optimal routing of electric vehicles in the setting of flexible edge cost functions and arbitrary initial states.

More precisely, the introduced state-based routing problem is a consolidated model covering energy-efficiency and time-dependency. Given two vertices and an initial state the routing problem is to find optimal paths yielding minimal final states, while the profile routing problem is to find optimal paths for all initial states. A universal method for applying shortest path techniques to profile routing is developed. To show the genericity and efficiency of this approach it is instantiated for two typical shortest path algorithms, namely for A* and Contraction Hierarchies. Especially using the latter, a highly efficient solution for energy-efficient profile routing is obtained.

Keywords: State-based, energy-efficient, time-dependent, flexible routing, profile search, electric vehicles.

1 Introduction

Computing energy-efficient paths for electric vehicles is of particular interest because of their limited range and long recharge periods. Shortest path algorithms can be used to provide a user with optimal driving directions, but the question is how to adapt efficient algorithms to respect the requirements of an accurate model of the vehicle and its environment.

Dijkstra's algorithm [4] is probably the most known shortest path algorithm, with A* being a natural extension using a heuristic lower bound on shortest path distances [9]. There are numerous extensions to the simple shortest path problem, two prominent examples of interest here are time-dependent routing described e.g. by Delling and Wagner [3] and energy-efficient routing described by Sachenbacher et al. [11]. Both of these problems are based on the idea of having an initial state, i.e. the departure time or the initial battery charge, and functions mapping the current state to edge costs, but they do not explicitely consider finding paths for all initial states.

Many different shortest path techniques were developed reducing the query time in a trade for preprocessing time and space requirements, a thorough review classifying those techniques is given by Bast et al. [1]. A prominent example of such an algorithm is Contraction Hierarchies (CHs) introduced by Geisberger [7].

R.C.-H. Hsu and W. Shangguang (Eds.): IOV 2014, LNCS 8662, pp. 21–30, 2014.

The problem is, that those techniques cannot be directly applied to solve state-based routing problems or profile queries. While the concept of CHs was adapted to time-dependent routing by Batz et al. [2] and to energy-efficient routing by Eisner et al. [5], this was done using a clever trick, which is applicable in these two specific settings: One can use backwards reachability to label all relevant edges and then run a simple routing algorithm such as Dijkstra's on the labeled edges. However, no general solution is given by these two approaches.

The routing problem considering all initial states is called *profile query* or *profile search* and was mentioned in the context of time-dependent routing [3]. It is a particularly interesting problem, because the initial state is often not known precisely, maybe due to inexact measurements of an electric vehicle's battery, or because it is configurable – for instance by charging the battery a little more before starting. Another reason for searching profiles might be the dependency on other problems, for example when routing is done in the context of vehicle scheduling problems, where different initial states are of interest. We expect this to be of particular interest for intermodal mobility, i.e. the combination of different modes of transportation, with Masuch et al. describing a promising example [10].

This paper introduces a powerful framework by adapting efficient state-of-the-art shortest path algorithms to find profiles describing optimal solutions for all initial states in a generic network with flexible edge weight functions. It is organized as follows. Section 2 introduces the state-based routing problem and the theory of profiles. Energy-efficient routing is shown to be one instance of this model. The profile routing problem is introduced and a theorem connects the routing problem to profile queries. Section 3 covers the generic approach to adapting shortest path algorithms to the profile routing problem. This approach is presented for a typical algorithm, namely A*. For the evaluation in Section 4, we also implemented Contraction Hierarchies and adapted them in the same way. Section 5 concludes with an interpretation and a discussion of open problems.

There are various approaches to introduce and combine different kinds of optimization criteria. The bicriteria search is probably the most known approach for finding Pareto-optimal paths, described for example by Skriver and Andersen [13]. It is similar to the profile search, because both problems use partially ordered values, but the state-based approach is more general and shortest path techniques can still be applied.

Another approach is to melt multiple criteria into a single criterion, for example by a linear function as described by Geisberger et al. [6]. The coefficients are meant to be dynamic, such that the precomputation of shortest path techniques must be valid for all coefficients. We consider a more general approach, where the coefficients of linear cost functions might be incorporated in the state, but probably losing the possibility of exploiting the special nature of linear functions – that is, handling flexible edge cost functions which are known to be linear is probably easier than handling non-linear functions.

2 Modelling

We define state-based routing and show energy-efficient routing to be an instance interesting for electric mobility. A formal definition of the profile routing problem is then based on defining routing profiles and profile preorders.

2.1 State-Based Routing

Instead of weighting edges with fixed and totally preordered costs, we use *monotone* ($s_1 \leq_S s_2 \to f(s_1) \leq_S f(s_2)$) and *extensive* ($s \leq_S f(s)$) state functions $S \to S$ on an arbitrary set of states S partially preordered by $\leq_S$ ensuring reflexivity and transitivity. If an edge's function $f : S \to S$ is undefined for some $s \in S$, written $f(s) = \bot$, it is not traversable for that state. We define $\bot \leq_S s$ for all states $s \in S$. Notice, that these functions are not necessarily commutative with respect to functional composition. We use the composition order $(f \circ g)(x) = g(f(x))$ to reflect how edges are concatenated to form a walk. A walk is a sequence of vertices, while a path is a cycle-free walk.

Definition 1. *The tuple* $(G, S, \leq_S, \mathcal{W})$ *is called a* state-based network*, if*

- $G = (V, E)$ *is a directed graph,*
- $(S, \leq_S)$ *is a partially preordered set of states,*
- $\mathcal{W} : E \to (S \to S)$ *is a weighting of monotone and extensive state functions.*

We extend $\mathcal{W}$ *to walks by edge-wise functional composition* $\mathcal{W}(\gamma) = \mathcal{W}(v_0, v_1) \circ \ldots \circ \mathcal{W}(v_{k-1}, v_k)$ *for all walks* $\gamma = (v_0, \ldots, v_k)$ *of* G*. For the* initial state $s \in S$*, we call* $\mathcal{W}(\gamma)(s) \in S$ *a* final state *with respect to walk* γ *and state* s*.*

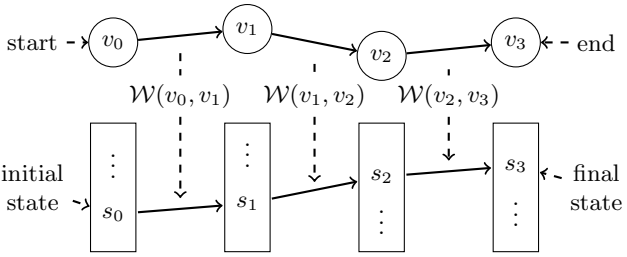

Fig. 1. A walk of four vertices (v_0, v_1, v_2, v_3) is shown as part of a state-based network. Every edge (v_i, v_{i+1}) is labeled with a state function $\mathcal{W}(v_i, v_{i+1}) : S \to S$. The state changes as we move along the walk: $s_{i+1} = \mathcal{W}(v_i, v_{i+1})(s_i) = \mathcal{W}(v_0, \ldots, v_{i+1})(s_0)$.

The following definition of the state-based routing problem requires two things to notice: Both monotonicity and extensivity is preserved under composition and for all walks there is at least one path (without cycles) yielding equivalent or better final states.

Definition 2. *Given a state-based network* $(G, S, \leq_S, \mathcal{W})$*, a start* $x \in V$*, a destination* $y \in V$ *and an initial state* $s \in S$*, the* state-based routing problem *is to find corresponding paths* π *from* x *to* y *for each minimal final state* $\mathcal{W}(\pi)(s)$ *except for equivalence.*

2.2 Energy-Efficient Routing

Finding energy-efficient routes is interesting for reasons of saving fossil fuels and anthropogenic carbon dioxide, but it is also of particular interest for users of electric vehicles which have a short cruising range and slow recharge rates. We want to determine if the battery charge is sufficient to reach the destination and which route we should take in order to retain most of the battery's energy. There is an algorithmic approach to this problem adapting A* using vertex potentials naturally given by their altitudes [11]. This example is used for evaluating our approach.

Energy-efficiency comprises various aspects, but we focus on battery constraints for electric vehicles. Given a battery capacity $K > 0$, we describe battery charges with $J \in [0, K]$. Energy is consumed as we travel along the edges of the road network, but by recuperation the battery status may also increase while losing potential energy. We consider recuperation from driving downhill, so we incorporate the altitude map in the form of potential energy levels. Battery constraints play a role whenever recuperation reaches the battery's capacity and whenever the battery is exhausted.

We can model this problem using a state-based network in the following way. The energy state is a pair of battery charges and potential energy levels $S = [0, K] \times \mathbb{R}$. The preorder $\leq_S$ is given by comparing the sums, i.e. $(J, h) \leq_S (J', h')$ if and only if $J + h \geq J' + h'$ (the sum of both values could also be used, but we want to keep the battery's state unbiased). Notice, that the comparison is inverted because of maximizing the remaining energy (instead of minimizing costs). Using potential energy as vertex potentials originates from [11]. We denote the potential energy by h_v for vertex $v \in V$.

We define a battery constraint $\mathcal{B}_{a,b}$ as a soft upper bound $b \in \mathbb{R}$ to represent fully charged batteries and a hard lower bound $a \in \mathbb{R}$ to represent empty batteries:

$$\mathcal{B}_{a,b}(x) = \begin{cases} b & \text{if } x > b, \\ x & \text{if } a \leq x \leq b, \text{and} \\ \bot & \text{if } x < a. \end{cases}$$

We can now define battery functions $S \to S$ from a vertex v to a vertex w with potentials h_v and h_w by

$$(J, h_v) \mapsto (\mathcal{B}_{a,b}(J - c - \Delta), h_w),$$

where $0 \leq a \leq b \leq K$ describe battery constraints, $c \in [0, K]$ describes the costs and $\Delta = h_w - h_v$ describes the difference of potentials. The function is undefined, if the second parameter is not equal to the potential h_v of v. Notice, that these functions are extensive, because $\mathcal{B}_{a,b}(J - c - \Delta) + h_w \leq J - c - \Delta + h_w = J - c + h_v \leq J + h_v$. The functions are also monotone due to the monotonicity of the battery constraint $\mathcal{B}_{a,b}$.

2.3 Profiles

While state-based routing problems ask for optimal paths given an initial state, we might also ask for a minimal set of optimal paths covering all initial states. This idea is adapted from time-dependent routing, where it is called a profile query [3]. The motivation behind profile routing is twofold: providing the user with more information and enabling bidirectional searches.

A profile shall be described by a set of walks sharing common end vertices, i.e. the same start and end vertex. At the same time, those profiles shall represent partially preordered values. The reason for that distinction is, that there may be values not representing any walks. This corresponds to the concept of heuristics in A* and in ALT using landmarks [8].

Two operations, namely concatenation $\circ$ and combination $\cup$, are essential. The former means to connect two consecutive profiles to form a new profile containing all pair-wise composed walks. The latter means to join both sets of walks to form a combined profile.

We define a partial preorder on profiles with the intended meaning that a profile may consist of more valuable walks than another profile. The imposed requirements are essential for proving the correctness of algorithms, but they also follow directly from the specification of state-based networks.

Definition 3. *Let $G = (V, E)$ be a graph, let $(M, \leq_M)$ be a partially preordered set of* profile values *and let $\circ, \cup : M \times M \to M$ be two operations on M, such that we can identify start and destination vertex of each profile $M \to V \times V$. We write $m_{x,y}$ in short to say that a profile m has start vertex x and destination vertex y. The partial preorder $\leq_M$ is called a* profile preorder *if all of the following conditions (for all resp. profiles) are satisfied:*

$$m_{x,y} \ \cup \ m'_{x,y} \leq_M m_{x,y} \tag{1}$$

$$m_{x,y}, \ m_{y,z} \leq_M m_{x,y} \circ m_{y,z} \tag{2}$$

$$m_{x,y} \leq_M m'_{x,y} \to \ m_{x,y} \ \cup \ m''_{x,y} \leq_M m'_{x,y} \ \cup \ m''_{x,y} \tag{3}$$

$$m_{y,z} \leq_M m_{y,z'} \to \ m_{x,y} \circ m_{y,z} \leq_M m_{x,y} \circ m_{y,z'} \tag{4}$$

$$m_{x,y} \leq_M m_{x',y} \to \ m_{x,y} \circ m_{y,z} \leq_M m_{x',y} \circ m_{y,z} \tag{5}$$

$$m_{x,y} \leq_M m_{x',y'}, m'_{x',y'} \to \ m_{x,y} \leq_M m_{x',y'} \ \cup \ m'_{x',y'} \tag{6}$$

When talking about profiles as sets of walks, the following definition helps understanding and handling profiles.

Definition 4. *Given a graph $G = (V, E)$, a* profile $m_{x,y} \subseteq \text{walks}_{x,y}$ *is a* set of walks *from x to y in G. The set of all profiles from x to y is denoted by $M_{x,y}$ (the power set of all walks from x to y) and the union of all profiles is denoted by $M := \bigcup_{x,y \in V} M_{x,y}$. Let $\leq_M$ be a profile preorder on M.*

- *The* combination $m_{x,y} \cup m'_{x,y}$ *of profiles $m_{x,y}, m'_{x,y}$ is the union of both sets.*
- *The* concatenation $m_{x,y} \circ m_{y,z}$ *of consecutive profiles $m_{x,y}, m_{y,z}$ is the set of element-wise concatenations $\{\gamma_1 \circ \gamma_2 \mid \gamma_1 \in m_{x,y}, \gamma_2 \in m_{y,z}\}$.*

- *A walk γ from x to y is said to* improve *(or is* not dominated by*) a profile $m_{x,y}$, if and only if $m_{x,y} \not\leq_M m_{x,y} \cup \{\gamma\}$. A profile $m'_{x,y}$* improves *$m_{x,y}$ if there is a walk $\gamma \in m'_{x,y}$ improving $m_{x,y}$.*
- *A profile $m_{x,y}$ is said to be* complete*, if and only if there is no walk γ from x to y improving $m_{x,y}$.*
- *A profile $m_{x,y}$ is said to be* minimal*, if and only if all walks $\gamma \in m_{x,y}$ improve $m_{x,y} \setminus \{\gamma\}$.*
- *A method* reduce *finds minimal subprofiles $m'_{x,y}$ of $m_{x,y}$, i.e. $m'_{x,y} \equiv_M m_{x,y}$ and $m'_{x,y}$ is minimal (not uniquely determined).*

In order to connect state-based routing and profiles the following definition introduces an induced profile preorder. Intuitively, a profile m_{x_1,y_1} is better than another profile m_{x_2,y_2}, if the walks in m_{x_1,y_1} yield better final states than all walks in m_{x_2,y_2} for all comparable initial states. We define that comparison formally as follows.

Definition 5. *Let $(G, S, \leq_S, \mathcal{W})$ be a state-based network and let M be the set of profiles, then $\leq_M$ is an* induced profile preorder *on M, where $m_{x_1,y_1} \leq_M m_{x_2,y_2}$ if and only if for all $s_2 \in S$ we have*

$$\forall \gamma_2 \in m_{x_2,y_2} : \mathcal{W}(\gamma_2)(s_2) = \bot$$

or else there is an $s_1 \in S$ with $s_2 \leq_S s_1$ and

$$\forall \gamma_2 \in m_{x_2,y_2}\ \exists \gamma_1 \in m_{x_1,y_1} : \mathcal{W}(\gamma_1)(s_1) \leq_S \mathcal{W}(\gamma_2)(s_2).$$

One of the main theoretical results of the paper is this relation and the following theorem. It is essential for connecting profile searches with the problem of finding optimal paths. It states, that routing problems in a state-based network always have an induced profile problem, which can be solved efficiently as described in the next section. The proof makes use of the state preorder $\leq_S$ and of the monotonicity and extensivity of weight functions, but must thoroughly account for the partiality of the weight functions and the genericity of states.

Theorem 1. *The induced profile preorder $\leq_M$ of a state-based network is a preorder and satisfies all conditions from Definition 3.*

After introducing profiles, which are the sets of paths contributing to optimal final states, we define a routing problem looking for minimal and complete profiles as follows.

Definition 6. *Given a state-based network $(G, S, \leq_S, \mathcal{W})$ and two vertices $x, y \in V$ the* state-based profile routing problem *is to find a minimal and complete profile $m_{x,y}$ with respect to the induced profile preorder $\leq_M$.*

For the sake of completeness, a solution to a state-based routing problem can be determined from the minimal and complete profile:

Lemma 1. *Given a solution $\pi \in \text{paths}_{x,y}$ of a state-based routing problem for a given initial state $s \in S$ and a solution $m \subseteq \text{paths}_{x,y}$ of a profile routing problem, then there is a path $\pi' \in m$ such that $\mathcal{W}(\pi)(s)$ is equivalent to $\mathcal{W}(\pi')(s)$.*

3 Algorithms

Shortest path algorithms explore the graph step by step labeling vertices with intermediate results. Because usually multiple queries need to be processed on the same graph, preprocessing the graph while preserving shortest paths is the key to improve query times.

The general idea for applying shortest path techniques to the profile routing problem is to replace comparison of values, addition of values and labeling of predecessors by their respective definitions on profiles, i.e. comparison, concatenation and combination of profiles.

In our experiments we use unbalanced binary trees, in which elements are inserted to the right subtree, if it is greater than the subtree's root, and to the left subtree otherwise.

As an example, the A* variant for profile searches is presented in Algorithm 1. The variables $m_{x,v} \in M_{x,v}$ are profiles for reaching vertex v from start x (similar to the predecessor variables of the original algorithm). In A* we additionally use heuristic lower bound values on the remaining distance denoted by $h_{v,y}$. We require $m_{x,v} \circ h_{v,y} \leq_M m_{x,y}$ for all $m_{x,v} \leq_M m_{x,y}$ representing a lower bound. In the case of a simple Dijkstra's algorithm, the heuristic profile values would be left out in line 4 and 5.

Algorithm 1. A* for Profile Routing

```
1  m_{x,v} ← ∅ for all v ∈ V
2  m_{x,x} ← {(x)}
3  queue q ← {x}
4  while m_{x,y} ≰_M m_{x,v} ∘ h_{v,y} for some v ∈ q do
5      remove v from q with minimal m_{x,v} ∘ h_{v,y}
6      for every successor w of v do
7          if m_{x,v} ∘ {(v,w)} improves m_{x,w} then
8              m_{x,w} ← reduce(m_{x,w} ∪ (m_{x,v} ∘ {(v,w)}))
9              add w to q
10 return m_{x,y}
```

Notice, that because of building solutions step by step only paths (walks without cycles) are constructed by this algorithm. This is important for the following theorem, an essential result of this paper:

Theorem 2. *Algorithm 1 computes a minimal and complete profile $m_{x,y}$ from x to y consisting only of paths (no cycles).*

This theorem can be proven by establishing a loop invariant and showing its initialization, its maintenance and its termination. Let π^v denote the prefix of a path π up to a certain vertex $v \in V$ (including v) and let π_v denote the suffix of a path π starting from a certain vertex $v \in V$ (including v). The loop invariant is: For each path π improving a (non-complete) profile $m_{x,y}$ or $m_{x,u}$ with $m_{x,y} \not\leq_M m_{x,u}$ there is a vertex v in π also contained in q, such that $\pi^v \in m_{x,v}$ and π^w improves $m_{x,w}$ for all consecutive successors w of v on π.

4 Evaluation

We evaluate our approach within the prototypic framework of Green Navigation, a tool for providing energy-efficient driving directions and for computing the range of electric vehicles. The project was initialized at the Technische Universität München and is continued at the University of Lübeck.

The energy-efficient routing problem and its profile variant is implemented based on the geospatial data taken from the collaborative OpenStreetMap (OSM) project and the altitude values taken from the NASA Shuttle Radar Topographic Mission (SRTM). A section representing Bavaria, a state of Germany with an interesting relief, is used in the experiments. A full charge allows a cruising range of about 150 km. For 1000 test cases the start and destination are chosen randomly (uniformly distributed among vertices) within reach. The experiments were carried out by the Java SE Runtime Environment on a single core of an Intel(R) Core(TM) i7-3520M CPU at 2.90GHz provided with 4 GB RAM.

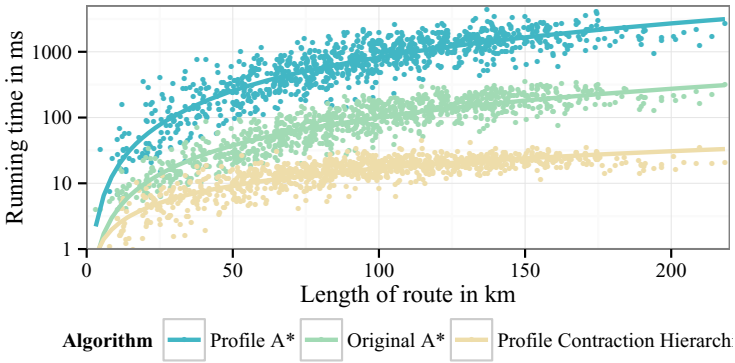

Fig. 2. The running times of different algorithms are shown on a logarithmic scale together with linear regressions of a logarithmic model

Figure 2 shows the running times for 1000 in-range test cases. The road graph contains 975,806 vertices and 1,885,037 edges. The original algorithm refers to the specialized A* version for energy-efficient routing [11] yielding an average running time of 281 ms. Profile searches are more complex, the adapted A* for profile searches is therefore slower with an average running time of 1106 ms.

Contraction Hierarchies were adapted in a similar way. The preprocessing of CHs takes approximately 15 minutes and contains 1,970,615 additional edges. The maximum profile size of the contracted graph is 6. Despite of solving a more difficult problem, the queries are much faster with an average running time of about 19 ms.

While A* performs reasonably good for profile routing, the CHs drastically improve the query time. This makes profile queries highly efficient for the energy-efficient routing problem. We are convinced, that these results can be significantly improved by using specialized data structures and sophisticated heuristics for the node ordering of CHs.

5 Conclusions

The essential idea of many routing problems is to maintain accumulated costs while traversing the network. The state-based routing problem is a generalization of the simple shortest path problem comprising a set of states and edge weight functions. The problem of finding solutions for all initial states is the profile routing problem. One interesting instance is energy-efficiency, but in the future we will show, that also time-dependent routing and even some variants of stochastic routing are instances.

One important aspect when considering instances of state-based routing is the descriptive complexity of the profiles. In case of the simple shortest path problem, profiles would be trivial and operations would be constant in time and space. In case of energy-efficient routing, concatenation does not add complexity while combination increases the complexity linearly.

We distinguish between profile values and profile walks in order to be able to use heuristics. In A^* a lower bound on the remaining path costs is computed usually using the straight-line distance, which does not represent an actual path. In the same way we use a profile value not representing an actual set of walks.

Many practical shortest path techniques trade preprocessing time and space for faster queries. However, it is not in general possible to apply those techniques to other routing problems. This paper defines the profile routing problem as a general class of routing problems that has a similar nature to the simple shortest path problem, such that shortest path techniques can be applied relatively easy, yet solving a more complex problem.

As a proof of concept, we instantiated our profile routing with A^* and contraction hierarchies and have shown its efficiency in the context of energy-efficient routing. Using Contraction Hierarchies we have now, to the best of our knowledge, obtained the currently most efficient profile routing system for the energy-efficient routing problem.

In the future, we want to analyze other aspects for state-based routing such as dynamic data, intermodality, fleet routing, congestion and also combinations of these. Furthermore, we want to integrate certain stochastic aspects, as described in our previous work [12] which is based on Uludag et al. [14]. We will show, that this can be done by using distribution functions of random variables as states and convolutions as state functions, which results also in interesting numerical challenges.

Acknowledgements. This work was supported by Gesellschaft für Energie und Klimaschutz Schleswig-Holstein (EKSH) GmbH and by the Graduate School for Computing in Medicine and Life Sciences at the University of Lübeck. We would like to thank our colleagues Christofer Krüger and Anne Reichart for their contributions.

References

1. Bast, H., Delling, D., Goldberg, A., Müller-Hannemann, M., Pajor, T., Sanders, P., Wagner, D., Werneck, R.: Route planning in transportation networks. Tech. rep., Microsoft Research (2014)
2. Batz, G.V., Delling, D., Sanders, P., Vetter, C.: Time-dependent contraction hierarchies. In: Proceedings of the 11th Workshop on Algorithm Engineering and Experiments (ALENEX 2009), pp. 97–105 (2009)
3. Delling, D., Wagner, D.: Time-dependent route planning. In: Ahuja, R.K., Möhring, R.H., Zaroliagis, C.D. (eds.) Robust and Online Large-Scale Optimization. LNCS, vol. 5868, pp. 207–230. Springer, Heidelberg (2009)
4. Dijkstra, E.W.: A note on two problems in connexion with graphs. Numerische Mathematik 1, 269–271 (1959)
5. Eisner, J., Funke, S., Storandt, S.: Optimal route planning for electric vehicles in large networks. In: Twenty-Fifth AAAI Conference on Artificial Intelligence, vol. 25, pp. 1108–1113 (2011)
6. Geisberger, R., Kobitzsch, M., Sanders, P.: Route planning with flexible objective functions. In: ALENEX, pp. 124–137. SIAM (2010)
7. Geisberger, R., Sanders, P., Schultes, D., Delling, D.: Contraction hierarchies: Faster and simpler hierarchical routing in road networks. In: McGeoch, C.C. (ed.) WEA 2008. LNCS, vol. 5038, pp. 319–333. Springer, Heidelberg (2008)
8. Goldberg, A.V., Harrelson, C.: Computing the shortest path: A search meets graph theory. In: Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 156–165. Society for Industrial and Applied Mathematics (2005)
9. Hart, P., Nilsson, N., Raphael, B.: A formal basis for the heuristic determination of minimum cost paths. IEEE Transactions on Systems Science and Cybernetics 4(2), 100–107 (1968)
10. Masuch, N., Lützenberger, M., Keiser, J.: An Open Extensible Platform for Intermodal Mobility Assistance. Procedia Computer Science 19, 396–403 (2013), http://www.sciencedirect.com/science/article/pii/S1877050913006625, the 4th International Conference on Ambient Systems, Networks and Technologies (ANT 2013), the 3rd International Conference on Sustainable Energy Information Technology, SEIT 2013
11. Sachenbacher, M., Leucker, M., Artmeier, A., Haselmayr, J.: Efficient energyoptimal routing for electric vehicles. In: AAAI Publications, Twenty-Fifth AAAI Conference on Artificial Intelligence, vol. 25, pp. 1402–1407 (2011)
12. Schönfelder, R., Leucker, M.: Stochastisches Routen für Elektrofahrzeuge. Energieinformatik (2011)
13. Skriver, A., Andersen, K.: A label correcting approach for solving bicriterion shortest-path problems. Computers & Operations Research 27(6), 507–524 (2000), http://www.sciencedirect.com/science/article/pii/S0305054899000374
14. Uludag, S., Uludag, Z., Nahrstedt, K., Lui, K.S., Baker, F.: A laplace transform-based method to stochastic path finding. In: IEEE International Conference on Communications, ICC 2009, pp. 1–5 (2009)