Energy-Efficient Routing: Taking Speed into Account

Frederik Hartmann and Stefan Funke

Institut für Formale Methoden der Informatik
Universität Stuttgart

Abstract. We introduce a novel variant of the problem of computing energy-efficient and quick routes in a road network. In contrast to previous route planning approaches we do not only make use of variation of the routes to save energy but also allow variation of driving speed along the route to achieve energy savings. Our approach is based on a simple yet fundamental insight about the optimal velocities along a fixed route and a reduction to the constrained shortest path problem.

1 Introduction

Taking energy consumption into account when planning a road trip becomes more and more of an issue. While in the old days, gasoline was cheap, today fuel prices are constantly increasing and due to the limited supply of fossil fuels probably always will be. While in this case taking energy consumption into account is mostly a means of saving money and reducing carbon dioxide emissions, in the context of electric vehicles (EVs) energy-aware route planning might even make the difference between reaching the destination and getting stranded halfway due to a depleted battery. In the medium term, EVs (in particular when recharged using renewable energies, e.g. from solar or wind power) are to replace fossil fuel driven vehicles, but currently the limited energy reservoir (typically around or less than 200 km cruising range) as well as the sparsity of battery reloading/switch stations makes energy awareness mandatory for EVs.

Artmeier et al. in [2] formalized one variant of energy-constrained route planning, taking into account special characteristics of EVs like energy recuperation. They essentially use a variant of the Bellman-Ford algorithm to compute the most energy-efficient route for an EV. Later Eisner et al. in [3] showed how to transform the problem instance such that Dijkstra's algorithm as well as the speedup technique of contraction hierarchies [6] apply. This resulted in data structures that can be precomputed in few minutes even on country-sized road networks like that of Germany allowing for query times several orders of magnitude faster than Bellman-Ford or Dijkstra. There is a fundamental disadvantage of this approach, though: it only aims at minimizing the energy consumption not taking into account travel time or length of the respective route. In particular, since the models in [2] and [3] always assume that road segments are used with their typical maximum speed (e.g. residential roads at 30 km/h, inner city

C. Lutz and M. Thielscher (Eds.): KI 2014, LNCS 8736, pp. 86–97, 2014.

roads 50 km/h, country roads 100 km/h, autobahns 130 km/h), applying these approaches often leads to routes which excessively use residential and inner city roads since at these lower speeds the energy consumption is minimal – routes of this type are hardly useful in practice.

In [7] Storandt considered the *constrained shortest path* (CSP) problem in the energy-aware routing context: for a given bound T on the travel time the goal is to compute the most energy-efficient path from some node s to some node t in the road network not exceeding travel time T (alternatively, one could also bound the energy-consumption and optimize the travel time). Typically one would query with a bound T which is – let's say – 10% above the time of the quickest path from s to t takes. The resulting paths are considerably more useful in practice since nobody is willing to take a route which takes twice as long as the quickest route just for some minuscule energy savings. One difficulty with this approach is the hardness of the CSP problem in general, even though for real-world problem instances, a carefully tuned implementation like the one in [7] can compute probably optimal results in reasonable time. A similar result in the context of bicycle routes has been presented in [8]. The approach by Funke and Storandt [5] on the other hand allows to optimize a conic combination of two or more criteria, e.g., energy consumption and travel time. While excluding some Pareto-optimal solutions, the preprocessing and the query – in contrast to the CSP approach – can be performed in polynomial time. Still, [7] and [5] assume that road segments are always traversed at their typical maximum speed, ignoring the fact that it is well possible to save some energy e.g. on an autobahn by going 100 km/h instead of 140 km/h.

Our Contribution

In this paper we introduce a novel variant of the problem of computing energy-efficient yet quick routes in a road network. In contrast to previous route planning approaches we do not only deliberately choose the routes to save energy but also allow variation of the driving speed along the route to achieve energy savings. The main insight which our approach is based upon is the fact that for *a fixed route and travel time* it is most energy-efficient to drive with uniform speed if possible. We propose efficient solution strategies to obtain solutions to the problem via reduction to several constrained shortest path (CSP) problem instances. In spite of CSP being NP-hard we can compute exact solutions for real-world problem instances within a reasonable amount of time.

2 Fundamentals and Basic Techniques

Before we can talk about actual algorithms we first need to introduce our mobility model. We assume the road network is given as a directed acyclic graph $G(V, E)$ with a straightline embedding in the plane. Each edge $e \in E$ has a road type $type(e) \in R$. Each road type $r \in R$ has an associated upper bound $v_{\max}(r)$ on the allowed speed on a road of that respective type. For example, for the road

type 'autobahn', we might have $v_{\max}(autobahn) = 150$; often we directly write $v_{\max}(e) = 150$ for an edge e of type 'autobahn'. By $|e|$ we denote the Euclidean length of an edge in the given straightline embedding.

Obviously, when travelling at speed v, it takes $|e|/v$ time to traverse a road segment e of length $|e|$. The energy consumption along a road segment has a more complicated dependence on the speed, though.

2.1 Energy Model

Air resistance is the main factor that influences energy/fuel consumption when the speed increases. This essentially leads to a quadratic dependence on the speed. We use the following function as proposed in [4]

$$f(v)[ml/m] : v \to c_1 v^2 - c_2 v + c_3 + \frac{c_4}{v}$$

where $c_1 = 0.00625968$, $c_2 = 0.11736$, $c_3 = 2.1714$, $c_4 = 18\frac{1}{3}$, see Figure 1. The speed v is expected to be given as $[m/s]$, the fuel consumption as $[ml/m]$. Above 54 km/h, this function is non-negative, convex and strictly increasing (in the E-mobility context, the minimum is already at a lower speed). Observe that it never pays off to drive more slowly than 54 km/h, and in case of edges e with $v_{\max}(e) < 54$km/h, it is most beneficial to use them at that maximum speed.

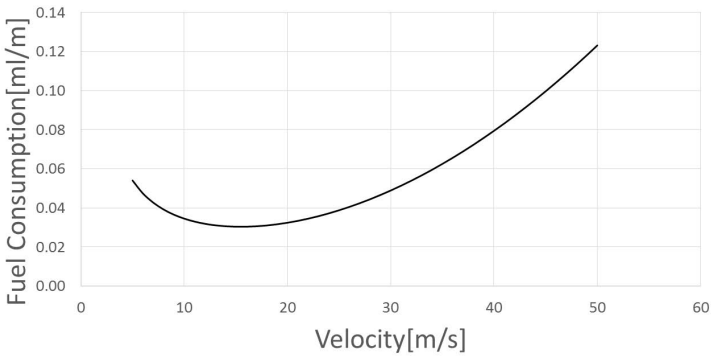


Fig. 1. Energy/Fuel Consumption dependent on velocity

2.2 Crucial Insight: Varying Speed Does Not Pay Off

Let us first examine how speed can be varied in different parts of a route if the travel time is to be kept constant. To that end consider without loss of generality a route section of unit length (not necessarily contiguous!) that is traversed in one unit of time and divide the route section into two halves of length 1/2 each.

Lemma 1. *If a unit length route section is traversed such that the speed in the first half is $v^{(l)}$, in the second half $v^{(r)}$, and the total travel time is one time unit, then for $v^{(l)} \leq v^{(r)}$ we have*

$$v^{(r)} = \frac{v^{(l)}}{2v^{(l)} - 1}$$

Proof. We have that $\frac{1/2}{v^{(l)}} + \frac{1/2}{v^{(r)}} = 1$, the above inequality follows. □

The following lemma essentially states that if we go 1 km/h slower on the first half, we have to go faster by more than 1 km/h in the second half:

Lemma 2. *If a unit length route section is traversed in one unit of time, once with speeds $v_1^{(l)}$ ($v_1^{(r)}$) in the left (right) half, once with speeds $v_2^{(l)}$ ($v_2^{(r)}$), and $1/2 < v_1^{(l)} < v_2^{(r)} \leq 1$, then we have $v_1^{(l)} + v_1^{(r)} > v_2^{(l)} + v_2^{(r)}$*

Proof. Lemma 1 implies that the dependence of $v^{(r)}$ from $v^{(l)}$ behaves like $x \mapsto \frac{x}{2x-1}$ for $x \in (0.5, 1]$ which has slope less or equal to 1. The lemma follows. □

We use the above two lemmas to prove the main theorem of this section.

Theorem 1. *Assume a unit length route section is traversed in one unit of time, once with speeds $v_1^{(l)}$ ($v_1^{(r)}$) in the left (right) half, once with speeds $v_2^{(l)}$ ($v_2^{(r)}$), and $1/2 < v_1^{(l)} < v_2^{((l))} \leq 1$. Then, the energy consumption of traversing the route section with speeds $v_1^{(l)}$, $v_1^{(r)}$ is strictly greater than the energy consumption of traversing the route section with speeds $v_2^{(l)}$, $v_2^{(r)}$ for strictly monotone and convex energy functions.*

Proof. Lemma 2 implies $v_1^{(l)} < v_2^{(l)} \leq 1 \leq v_2^{(r)} < v_1^{(r)}$. The energy consumptions to compare are $E_1 := (f(v_1^{(l)}) + f(v_1^{(r)}))/2$ and $E_2 := (f(v_2^{(l)}) + f(v_2^{(r)}))/2$.

Let $p_i^{(l)} := (v_i^{(l)}, f(v_i^{(l)})) \in \mathbb{R}^2$ and $p_i^{(r)} := (v_i^{(r)}, f(v_i^{(r)})) \in \mathbb{R}^2$. By convexity of f we know that the segment $p_2^{(l)}p_2^{(r)}$ lies below the segment $p_1^{(l)}p_1^{(r)}$ between $v_2^{(l)}$ and $v_2^{(r)}$. Lemma 2 also implies that $(v_2^{(l)} + v_2^{(r)})/2 < (v_1^{(l)} + v_1^{(r)})/2$, hence with the monotonicity of f we obtain $E_1 > E_2$. □

Essentially this theorem states that if a route section is to be traversed within a certain time, it is most energy-efficient to do so at a uniform speed if possible. This leads to the following corollary which lies at the very heart of our routing approach.

Corollary 1. *If for a route section s that has to be traversed in time t it is possible to traverse it at uniform speed $v := |s|/t$, this is the most energy efficient way of traversal.*

Proof. Assume the most energy efficient way of traversal is not at uniform speed. Then we can identify a subsection (potentially infinitesimally small) where the first half is traversed at speed $v^{(l)}$, the second half at speed $v^{(r)}$, w.l.o.g.

$v^{(l)} < v^{(r)}$. Increasing $v^{(l)}$ a bit and hence being able to decrease $v^{(r)}$ without affecting the travel time for this subsection, we obtain a more energy efficient way of traversal according to our Theorem, contradicting the assumption that we started with the most energy efficient way of traversal. □

This corollary immediately implies that on road sections of the same type (uniform speed-limit, not necessarily contiguous), we should always drive with uniform speed.

Furthermore, if we have a route composed of different road sections (with differing maximum speeds), the energy-optimal traversal strategy if a fixed travel time t has to be met is the following:

- along the route the maximum speed of travel is v^*
- for all road sections e with $v_{\max}(e) < v^*$, we travel at speed $v_{\max}$
- for all road sections e with $v_{\max}(e) \geq v^*$ we travel at speed v^*

Again, correctness follows from the fact that for road sections (even non-contiguous) with the same maximum speed limit it is most beneficial to drive at uniform speed. If, there is some route section e where we drive at speed $v' < v_{\max(e)}$ and $v' < v^*$, it is more economical to drive slightly slower than v^* on some other route section and slightly faster on e according to our Theorem.

In summary, the above discussion implies that once we fix a maximum speed in our search for an energy-optimal s-t path with time bound T, the speeds on all edges (and hence both time-consumption as well as energy consumption) are uniquely determined, hence turning our problem into an 'ordinary' constrained shortest path problem where for two fixed metrics we are to compute the optimal path with respect to one metric satisfying a constraint on the second.

Furthermore observe that due to the fact that road sections of the same type have to be traversed at the same speed, the energy/travel time characteristics of a path are solely determined by the aggregated lengths of the different road type sections (e.g. 20 km inner-city road, 80 km country road, 400 km autobahn).

2.3 Contraction Hierarchies (CH)

Our proposed algorithms make use of a very elegant yet effective speedup technique for shortest path queries which we will sketch briefly in the following. The CH preprocessing technique as introduced by Geisberger et al. in [6] augments the graph $G(V, E)$ with a set E' of so called shortcuts, which span (large) sections of shortest paths and hence allow for a reduction of edge relaxations and node settling steps in a Dijkstra run. During the CH-construction nodes are contracted one-by-one in some suitable order, thereby assigning a label $l : V \to \mathbb{N}$ to each node. An edge (v, w) is referred to as upwards if $l(v) < l(w)$ and downwards otherwise, a path is called upwards/downwards if it consists exclusively of edges of that type. Shortcuts are inserted in such a way, that for each source-target pair $s, t \in V$ a shortest path exists in $G' = G(V, E \cup E')$ which can be decomposed into an upward section starting at s followed by a downward section ending in t. This property allows to restrict a bidirectional Dijkstra run

to $G^{\uparrow}_{out}(s)$ and $G^{\downarrow}_{in}(t)$ which refer to the subgraphs of G' containing only all upwards paths starting in s or all downwards paths ending in t respectively. For 'normal' shortest path queries, the restriction to $G^{\uparrow}_{out}(s)$ and $G^{\downarrow}_{in}(t)$ improves query times by several orders of magnitude.

2.4 The Constrained Shortest Path Problem

The two-criteria Constrained Shortest Path problem (CSP) can be defined as follows: We are given a (di)graph $G(V, E)$ ($|V| = n, |E| = m$), a cost function $c : E \to \mathbb{R}^+$ and a resource consumption function $r : E \to \mathbb{R}^+$ on the edges. A query is specified by a source and a target node $s, t \in V$ as well as a resource bound $R \in \mathbb{R}^+$. The goal is to determine the minimum cost path from s to t whose resource consumption does not exceed R.

In our context of energy-efficient routing, the cost function equals energy consumption, and the resource consumption equals travel time. It is important to note, though, that CSP assumes *scalar values* for the cost and resource values of individual edges, whereas in our case part of the problem is determining the optimum speed for each road segment.

Typical solutions to the CSP problem enumerate Pareto-optimal solutions by for example label setting [1]. In spite of NP-hardness of the CSP problem in general, these approaches are somewhat feasible for real-world data, in particular when combined with speedup techniques like contraction hierarchies, e.g. [7].

Our approach of taking into account speed variations into the energy-optimization problem will use (CH-accelerated) CSP as a basic building block.

2.5 The General Energy-Efficient Routing Problem

With our insights from Section 2.2 we are now ready to formally define the *General Energy-Efficient Routing Problem (GEERP)*:

Definition 1 (GEERP). *Given the straightline embedding of a directed graph $G(V, E)$, maximum speeds $v_{\max} : E \to \mathbb{R}$ on the edges, and an energy consumption function $f : \mathbb{R} \to \mathbb{R}$, the goal is to determine for a query (s, t, T) a path π from s to t as well as a speed value v^* such that*

- *travel time along π does not exceed the time bound T:*

$$\sum_{e \in \pi} \frac{|e|}{\min(v_{\max}(e), v^*))} \leq T$$

- *the energy consumption*

$$\sum_{e \in \pi} |e| \cdot f(\min(v_{\max}(e), v^*))$$

is minimal over all possible paths and speed values.

The assumption that one drives either at speed v^* or $v_{\max}(e)$ on an edge e of the path seems to be restrictive at first, but the above discussion has hopefully made clear that an optimal solution to the problem must have this special structure, so in fact we are also taking care of the general problem where arbitrary speeds are allowed on the road segments along the path π.

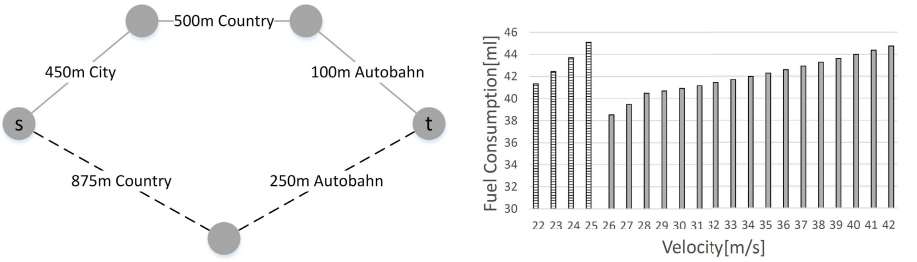
3 Algorithms

In the following we will describe algorithms to solve GEERP using the insights of the previous Section. For simplicity let us first assume that the possible speed values are *discrete*. We will come back to that issue later again.

3.1 Simple Reduction to CSP

If we have a discrete set of speeds $\{v_1, v_2, \dots v_k\}$, we can 'simply' solve GEERP by considering k CSP instances; the i-th CSP instance is constructed by setting the travel times and energy consumption according to a speed of v_i (or the respective maximum speed if that lies below v_i). Among the at most k feasible CSP solutions we pick the one with minimum energy consumption.

Note, though, that binary search over the set of feasible speeds is not possible, since the minimum energy consumption to get from s to t within a certain time limit does not depend monotonically on the speed, see Figure 2.



(a) Street Graph (b) Optimal CSP Solutions

Fig. 2. Non-monotonical energy consumption with two alternative routes and a time constraint of 55 seconds (speed limits as in Table 1). Up to 25 m/s the lower path yields the optimal fuel consumption.

3.2 SpeedUp Using CH

The running times to solve the CSP problem on real-world roadmap data can be drastically improved by employing the speedup technique of contraction hierarchies. For example in [8], a contraction hierarchy is constructed which guarantees

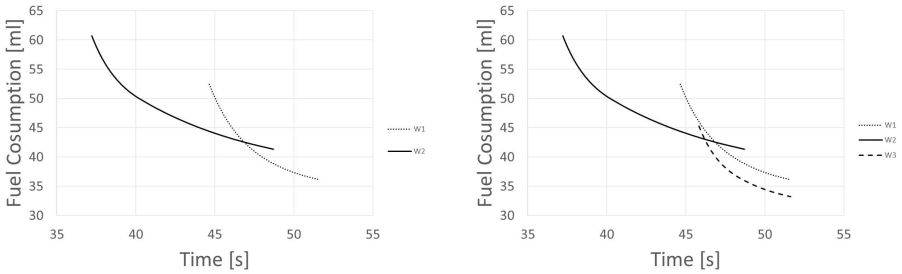
to preserve all Pareto-optimal paths. The crucial operation in the course of the CH construction is a *node contraction*. In [8], whenever a node v with neighbors u, w and $(u, v) \in E$, $(v, w) \in E$ is to be contracted, a shortcut (u, w) (with cost/resource consumption of the path uvw) has to be created if uvw is a Pareto-optimal path (note that this – in contrast to the single-metric case – typically leads to multi-edges between nodes). We could transfer this idea in a straightforward manner by constructing for each speed v_i a respective contraction hierarchy and use this for answering of the respective CSP instance. This seems ineffective, though, since we expect a lot of correlation between 'good' paths in the different contraction hierarchy graphs in particular for only slight variations of the speed. The space consumption of this approach is prohibitive: essentially we would need to store k times the CH-augmented road network (here k is the number of different velocities to be considered) – unrealistic for county-sized networks like that of Germany.

Instead our approach is to construct *one* contraction hierarchy that can be used for all CSP instances to be solved during the course of our algorithm. Again, the crucial operation is the node contraction step. When contracting a node v with neighbors u, w and $(u, v) \in E$ and $(v, w) \in E$, we need to create a shortcut uw if there exists a velocity s^* for which uvw is Pareto optimal. So we have to compare the velocity-dependent functions associated with paths(which have the same source and target) to decide on the creation of a shortcut. In Figure 3, left, we have depicted two paths with their velocity/fuel-consumption dependency. Each of the paths W1 and W2 is optimal for some choices of velocity. On the other hand, Figure 3, right, depicts three paths W1, W2, W3, where W1 is dominated by W2 and W3 since there is no speed value where W3 is on the boundary of the lower envelope. To decide whether a shortcut replacing uvw has to be created, we need to determine for all possible velocities the most energy-efficient paths from u to w; if uvw is amongst them the shortcut is necessary. We do this by explicitly computing the lower envelope of the speed-vs-energy function of all paths from u to w. Note that while the complexity of the speed-vs-energy function of a single route is constant (due to constantly many road types only), the lower envelope might exhibit almost arbitrary complexity. If during the CH construction the complexity of the lower envelope for some u, w-pair explodes, we refrain from contracting node v. In general – as for example in [8] – we only contract about 99.3% of all nodes leaving the remaining uncontracted.

In our experimental Section we will see that while we are far away from the speedups possible for 'ordinary' shortest path queries (see [6]), our CH constructed on edge cost functions still yields an order of magnitude improvement compared to unaccelerated CSP constructions.

3.3 Heuristic Solutions

Even the fastest exact solution is bound by the possibly exponential-sized Pareto-optimal solution sets, therefore a fast and simple solution to the CSP instances (and hence GEERP) cannot be expected. For many real world applications like



(a) Pareto-optimal Set after Insertion of W1 and W2

(b) Set after Insertion of Path W3 with now obsolete Path W1

Fig. 3. Two dimensional Example for combined Domination of Paths

the energy efficient routing a really optimal solution is not required, since a good approximation will already lead to a substantial improvement.

If we only consider a solution to be Pareto-optimal if it improves the target function by at least $x\%$ we are able to reduce the size of the solution sets substantially in real world applications. Even though the theoretical approximation guarantee is only $(1+x)^{(i-1)}$ with i being the edge count in the optimal path, in practice, the resulting solutions are very close to optimal.

3.4 Further Remarks

At the very beginning we have restricted ourselves to a *discrete* set of velocities. Intuitively, this should not make a big difference, but we need to convince ourselves that this is indeed the case. Consider our energy cost function and an optimal path π^* from s to t with maximum velocity v^*. Clearly, we can also traverse the same path with the 'next' velocity $v' > v^*$ present in our discrete set of velocities using at most a factor of $\frac{f(v')}{f(v^*)}$ more energy. If we discretize the velocities at a granularity of at most 1 m/s, the respective difference cannot exceed 5.1%; a finer discretization, of course, leads to a reduction of the provable error (e.g. 0.5m/s yields at most 2.1%).

4 Experimental Results

To validate our approach and obtain approximation estimates, several real world graphs from OpenStreetMap were chosen as test instances. We used Java 7 as programming language with OpenJDK 7 as runtime environment. The contraction hierarchy was computed on a server with two AMD Opteron 6172 CPUs and 96 GB RAM. All queries were performed on a workstation with one Core i7 4770K CPU and 16GB RAM.

4.1 Benchmark Instances

The edge types shown in Table 1 represent the main types of streets found in Germany with their usual speed limitations. Note that our observations in Section 2.2 can easily be extended to the case where minimum speeds are given. In this case, if v^* is smaller than $v_{\min}(e)$, e is traversed at speed $v_{\min}(e)$.

Table 1. Used Street Types

	City	Country	Autobahn
v_{min}[km/h]	50	80	100
v_{max}[km/h]	50	100	150
v_{min}[m/s]	14	22	28
v_{max}[m/s]	14	28	42

Table 2. Size and Area of used Graphs

Graph	Area	Nodes	Edges
DE	Germany	19,478,240	39,454,253
BW	Baden-Württ.	2,911,711	5,903,801
ST	Stuttgart	924,688	1,876,030

We created the graphs shown in Table 2 by mapping the OSM edge types to their closest representative. Edge types that can't be used by motorized vehicles or requiring special permits have been removed.

4.2 CH-Creation

Table 3 shows the statistics for the CH construction. As can be seen, the number of edges has increased by less than factor of 2 after contraction of 99.3% of the nodes, the remaining 0.7% *peak nodes* were left uncontracted. As observed previously, e.g. in [7], contracting all nodes typically leads to a huge number of shortcuts and even worse query performance.

Table 3. Size and preprocessing Time of CH Creation (using 20 cores on our server)

Graph	Nodes	Edges	Peak Nodes	Preprocessing Time[s]
DE-CH	19,478,240	70,128,496	154,856 (0.7%)	11,174
BW-CH	2,911,711	10,972,106	20,317 (0.7%)	4,767
ST-CH	924,688	3,452,011	2,743 (0.7%)	2,408

4.3 CH-Accelerated CSP Queries – Exact and Approximate

We compared the runtime and number of priority queue pulls of the standard label-setting CSP algorithm with the CH-label-setting algorithm. As to be expected we observe an improvement by an order of magnitude, see Table 4.

With these query times being considerably above the response times desirable for route planning, we applied the heuristic proposed in Section 3.3 to reduce the runtime while preserving a good solution quality. As shown in Table 5, the runtime improves dramatically while keeping the average error below the required minimal Pareto difference. If we accept an average error of 0.4%, a CSP Query in BW can be answered in about 700 ms. Even on a country-sized road network like that of Germany, we achieve query times of few seconds.

Table 4. Query times for label-setting und CSP-CH at 42 m/s for the most energy-efficient path using at most 20% more time than the quickest path. Average of 1,000 (100 for BW/BW-CH) queries.

Graph	label-setting		CSP-CH	
	Query[ms]	Pulls	Query[ms]	Pulls
BW / BW-CH	331,143	$6.3 \cdot 10^8$	17,911	$1.8 \cdot 10^7$
ST / ST-CH	27,010	$8.2 \cdot 10^7$	1,730	$2.4 \cdot 10^6$

Table 5. Heuristic CSP solution; pruning ratio for Pareto sets, average (ΔE) and maximum ($\Delta E_{\max}$) deviation from energy consumption of optimal solution. 'Unused' time budget (ΔT). 42 m/s; at most 20% above quickest path.

	ST-CH				BW-CH				DE-CH
Pruning ratio	query time (ms)	ΔE	ΔE_{max}	ΔT	query time (ms)	ΔE	ΔE_{max}	ΔT	query (ms)
1	1,500	-	-	-	25,900 (100%)	-	-	-	570,691
1.001	416	0.009%	0.06%	−0.017%	4,100 (16%)	0.015%	0.08%	−0.016%	77,300
1.005	146	0.114%	0.91%	−0.174%	1,200 (5%)	0.152%	0.85%	−0.136%	20,200
1.01	103	0.300%	1.512%	−0.321%	692 (3%)	0.375%	1.83%	−0.405%	8,900

4.4 Solving GEERP

To solve the actual GEERP instance, we proceed as follows: First, we compute for every possible speed a rough solution using a rather coarse pruning rule for the Pareto sets. We start with the highest possible speed decreasing as long as feasible (i.e. within the given time bound T) solutions are found. This initial run identifies a smaller set of velocities where a more fine-grained approximation is computed. Table 6 lists the results.

Table 6. Heuristic GEERP solution results for DE-CH with total query time, preselection time and average cardinality of the preselected velocity set. The preselected velocity set was obtained by performing a coarse approximation and selecting velocities around the best solution. Average of 100 queries.

Fine-Grained Approximation	Preselection Pruning Rule	Query Time[s] (including Preselection)	Preselected Velocities
1.005	1.05	106.7 (30.3)	7.6
1.005	1.025	80.7 (49.1)	4.4
1.005	1.01	126.2 (109.6)	2.1
1.005	-	196.0	-
1.001	1.05	327.2 (30.3)	7.6
1.001	1.025	175.6 (49.1)	4.4
1.001	1.01	325.3 (109.6)	2.1
1.001	-	1,009.8	-

In summary, we can compute a very good approximation to the most energy-efficient route for given source and target within Germany meeting a hard time constraint within a few minutes. As an example, we considered a source-target pair in Germany with a minimum travel time of 3,815 seconds and a fuel consumption of 9,025ml. If we accepted a travel time of 4,570 seconds, a path could be found by using a maximum speed CSP that required 6,825ml of fuel. By applying our GEERP solution, a path could be found that only required 6,098ml fuel, an improvement of about 25% compared to the shortest path and an additional improvement of 10% compared to the CSP path.

References

1. Aggarwal, V., Aneja, Y., Nair, K.: Minimal spanning tree subject to a side constraint. In: 32nd ACM Symposium on Theory of Computing (STOC), pp. 286–295 (1982)
2. Artmeier, A., Haselmayr, J., Leucker, M., Sachenbacher, M.: The shortest path problem revisited: Optimal routing for electric vehicles. In: Dillmann, R., Beyerer, J., Hanebeck, U.D., Schultz, T. (eds.) KI 2010. LNCS, vol. 6359, pp. 309–316. Springer, Heidelberg (2010)
3. Eisner, J., Funke, S., Storandt, S.: Optimal route planning for electric vehicles in large networks. In: AAAI Conference on Artificial Intelligence (2011), http://www.aaai.org/ocs/index.php/AAAI/AAAI11/paper/view/3637
4. Freie Universität Berlin, Institut für Chemie und Biochemie: Kfz energetisch betrachtet (January 2009), http://www.chemie.fu-berlin.de/chemistry/general/kfz-energetisch.html
5. Funke, S., Storandt, S.: Polynomial-time construction of contraction hierarchies for multi-criteria objectives. In: Algorithm Engineering and Experiments (ALENEX), pp. 41–54 (2013)
6. 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), http://portal.acm.org/citation.cfm?id=1788888.1788912
7. Storandt, S.: Quick and energy-efficient routes: Computing constrained shortest paths for electric vehicles. In: Proceedings of the 5th ACM SIGSPATIAL International Workshop on Computational Transportation Science, IWCTS 2012, pp. 20–25. ACM, New York (2012)
8. Storandt, S.: Route planning for bicycles - exact constrained shortest paths made practical via contraction hierarchy. In: 22nd Int. Conf. on Automated Planning and Scheduling, ICAPS (2012)