Efficient Energy-Optimal Routing for Electric Vehicles

Martin Sachenbacher
Technische Universität München
Department of Informatics
Boltzmannstraße 3
85748 Garching, Germany
sachenba@in.tum.de

Martin Leucker
Universität zu Lübeck
Institut für Softwaretechnik
Ratzeburger Allee 160
23562 Lübeck, Germany
leucker@isp.uni-luebeck.de

Andreas Artmeier
Julian Haselmayr
Technische Universität München
Department of Informatics
Boltzmannstraße 3
85748 Garching, Germany
{artmeier,haselmay}@in.tum.de

Abstract

Traditionally routing has focused on finding shortest paths in networks with positive, static edge costs representing the distance between two nodes. Energy-optimal routing for electric vehicles creates novel algorithmic challenges, as simply understanding edge costs as energy values and applying standard algorithms does not work. First, edge costs can be negative due to recuperation, excluding Dijkstra-like algorithms. Second, edge costs may depend on parameters such as vehicle weight only known at query time, ruling out existing preprocessing techniques. Third, considering battery capacity limitations implies that the cost of a path is no longer just the sum of its edge costs. This paper shows how these challenges can be met within the framework of A* search. We show how the specific domain gives rise to a consistent heuristic function yielding an $O(n^2)$ routing algorithm. Moreover, we show how battery constraints can be treated by dynamically adapting edge costs and hence can be handled in the same way as parameters given at query time, without increasing run-time complexity. Experimental results with real road networks and vehicle data demonstrate the advantages of our solution.

Introduction

Due to their efficiency and their ability to run on regenerative energy sources, electric vehicles (EV) that are partially or fully powered by batteries will significantly shape the road traffic of the future. However, because of limited battery capacities and long recharge times, techniques for energy-optimized driving and accurate prediction of remaining cruising range are even more important for such EVs than they are for conventional vehicles.

The goal of energy-efficient driving of EVs creates novel algorithmic challenges for navigation systems and route planners. Traditionally, routing has focused on finding shortest paths in networks with positive, static edge costs that represent the distance between two nodes. The best known algorithm for this case is Dijkstra (Dijkstra 1959) with time complexity $O(n^2)$. State-of-the-art route planning combines this algorithm with graph preprocessing techniques like contraction hierarchies (Geisberger et al. 2008), highway hierarchies (Sanders and Schultes 2005) and transit vertex routing (Bast et al. 2007).

Figure 1: Prototype of our energy-optimal route planner.

However, for energy-optimal EV routing, simply understanding edge costs as energy values and applying these standard algorithms does not work. First, part of the efficiency of EVs results from their ability to recover some of their kinetic and/or potential energy during deceleration phases. This so-called recuperation or regenerative braking means that edge costs can be negative, which excludes straightforward application of greedy Dijkstra-like algorithms. Second, edge costs may be complex to compute and may depend on a number of parameters such as vehicle payload, auxiliary consumers, etc. only known at query time. This rules out existing preprocessing techniques and techniques such as Johnson's algorithm (Johnson 1977), which is based on global graph analysis to eliminate negative edges. Third, considering battery capacity limitations means that additional energy losses or gains can arise from taking a specific path; for example, if the battery is already fully charged when entering a negative edge, recuperation is no longer possible. This implies that the cost of a path is no longer just the sum of its edge costs, as some costs cannot be statically attributed to individual edges. While extensions of the shortest path problem to incorporate such additional constraints exist (Joksch 1966), they are in general known to be NP-complete (Garey and Johnson 1979).

In this paper, we propose a solution for energy-optimal routing for EVs that meets the above challenges within the framework of A* search (Hart, Nilsson, and Raphael 1968).

The key idea of the approach is to exploit domain-specific knowledge to identify and bound different forms of energy, which gives rise to a consistent heuristic function and yields an $O(n^2)$ routing algorithm that expands only as few nodes as necessary. Moreover, we then show how battery constraints can be incorporated into this algorithm by dynamically adapting edge costs during the search; thus, they can be handled in the same way as parameters given at query time, without increasing the overall run-time complexity. Our solution improves by an order of magnitude upon previously published results (Artmeier et al. 2010) that presented an $O(n^3)$ algorithm for a simplified version of the problem. We implemented our approach within a prototypic route planning system for EVs (see Figure 1).

The remainder of the paper is structured as follows. In the next section, we introduce the problem of finding the most energy-efficient path for battery-powered EVs with recuperation in a graph-theoretical context. Then, we use a small example to present the idea of deriving a consistent heuristic for energy consumption by distinguishing and establishing bounds for two different forms of energy. Subsequently, we propose our algorithmic solution based on A* search and show how additional, path-related costs can be incorporated into this framework without compromising complexity and correctness. Finally, experimental results with a prototypic implementation using real road networks and vehicle data demonstrate the effectiveness of our solution.

Energy-optimal paths

We assume that a road network is given as a directed graph $G = (V, E)$ with $|V| = n$ and $|E| = m$. Vertices $v \in V$ represent points on the map and edges $e \in E$ represent connections between these points corresponding to road sections. We assume that for each vertex an elevation $z : V \to \mathbb{R}_0^+$ is given, and for each edge a length $l : E \to \mathbb{R}^+$ and a speed limit $s : E \to \mathbb{N}$ is given.

A path P is then a sequence of k vertices $(v_1, v_2, \dots, v_k)$ with $(v_i, v_{i+1}) \in E$ for $i = 1, 2, \dots, k$-1. We assume that along a path, the car is driven at the respective speed limit of each road section, and when transiting from one section (v_i, v_{i+1}) to the next one (v_{i+1}, v_{i+2}), adapting from $s(v_i, v_{i+1})$ to the new (possibly higher or lower) speed $s(v_{i+1}, v_{i+2})$ can be regarded as instantaneous.

In addition, we assume that further parameters that influence the vehicle's energy consumption, such as its mass and air drag coefficient, friction coefficients, etc. are available, albeit not necessarily beforehand.

Given this setting, we now consider the amount of energy consumed or gained by an EV when passing an edge in the network. We consider in our model two different forms of energy that can occur:

Potential energy E_P. We assume a function $E_P(z(a), \dots)$ models the potential energy resulting from elevation of a vertex. When traveling along an edge (a, b) in the path, the energy $c_P(a, b) = E_P(z(b), \dots) - E_P(z(a), \dots)$ has to be spent when $z(a) \leq z(b)$, resp. it can be (partly) recuperated by the EV when $z(a) > z(b)$.

Energy loss E_L. We assume a function $E_L(l(e), s(e), \dots)$ models loss of energy to the environment, for example due to rolling and aerodynamic resistance or conversion losses during recuperation, when passing an edge $e = (a, b)$. We assume this function is linearly increasing in $l(e)$ and monotonically increasing in $s(e)$. The energy $c_L(a, b) = E_L(l(e), s(e), \dots)$ cannot be recuperated by the EV.

For example, in our experiments we used the specific functions

$$c_P(a, b) = mg(z(b) - z(a))$$

where m is vehicle mass including payload, g is gravitational acceleration, and

$$c_L(a, b) = \begin{cases} \eta_r \cdot c_R(a, b) - c_P(a, b) & \text{if } c_R(a, b) \leq 0 \\ \frac{1}{\eta_c} \cdot c_R(a, b) - c_P(a, b) & \text{if } c_R(a, b) > 0 \end{cases}$$

where $\eta_c \in]0, 1]$ and $\eta_r \in [0, 1]$ are efficiency factors and $c_R(a, b)$ is the auxiliary term

$$c_R(a, b) = c_P(a, b) + f_r mgl(a, b) + \frac{1}{2}\rho A c_w s(a, b)^2 l(a, b)$$

with f_r being a friction coefficient, ρ the air density, A the vehicle's cross section area, and c_w its air drag coefficient.

The functions $c_P(a, b)$ and $c_L(a, b)$ sum up to an edge weight function $c : E \to \mathbb{R}$, $c(a, b) = c_P(a, b) + c_L(a, b)$ that models the amount of energy required to drive the road section (a, b). We call the weighted graph $G = (V, E, c)$ energy graph. Due to conservation of energy, no negative cycles exist in G.

Next, we consider the amount of energy needed by an EV to travel along a path P in the network. We call such a mapping from P to $\mathbb{R}$ a path cost function. Let $P = (v_1, v_2, \dots, v_k)$ be a path in G, and $P^i = (v_1, v_2, \dots, v_i)$ be the subpath of P ending in vertex v_i (with $i \leq k$), $P^k = P$. A first approach is to just sum the edge costs along the path. This is accomplished by the path cost function

$$c(P^k) = \begin{cases} 0 & \text{if } k = 1, \\ \Delta^k & \text{if } k > 1 \end{cases}$$

where $\Delta^k = c(P^{k-1}) + c(v_{k-1}, v_k)$. Finding a path P with minimal cost $c(P)$ is a classic shortest path problem, where some of the edges can have negative values.

However, a more realistic model has to include the EV's battery, which has an initial charge $J \in \mathbb{R}^+$. Paths are only feasible if the charge of the battery does not fall below zero. If we express infeasibility of paths by infinite costs, the following path cost function takes this constraint into account:

$$c_J(P^k) = \begin{cases} 0 & \text{if } k = 1, \\ \Delta^k & \text{if } k > 1 \text{ and } \Delta^k \leq J, \\ \infty & \text{if } k > 1 \text{ and } \Delta^k > J \end{cases}$$

where $\Delta^k = c_J(P^{k-1}) + c(v_{k-1}, v_k)$. In c_J, the cost of the path is the sum of the spent (or gained) energy, and infinity if this exceeds the battery charge along the way.

Finally, we also have to consider that the battery has a maximum capacity $C \in \mathbb{R}^+$, $J \leq C$. Recharging (at negative edges) is thus only possible as long as the battery has

sufficient free capacity (initially, $C-J$). The following more refined path cost function considers also this constraint:

$$c_{CJ}(P^k) = \begin{cases} C - J & \text{if } k = 1, \\ 0 & \text{if } k > 1, \Delta^k < 0, \\ \Delta^k & \text{if } k > 1, 0 \leq \Delta^k \leq C, \\ \infty & \text{if } k > 1, \Delta^k > C \end{cases}$$

where $\Delta^k = c_{CJ}(P^{k-1}) + c(v_{k-1}, v_k)$. In c_{CJ}, the cost of the path is equal to the remaining free battery capacity, and infinity if the battery charge is exhausted along the way.

Definition 1 (Energy-optimal routing problem) *Given an energy graph $G = (V, E, c)$, two vertices $s, t \in V$, an initial charge $J \in \mathbb{R}^+$ and a maximum capacity $C \in \mathbb{R}^+$, $J \leq C$, the energy-optimal routing problem is to find a path P in G from s to t with minimal path cost $c_{CJ}(P)$.*

Such energy-optimal routes correspond to paths which are feasible to use and where the remaining free battery capacity is minimal (equivalently, where the remaining battery charge at the end of the route is maximal).

Computing energy-optimal paths using A*

To simplify the presentation of our approach, we first elaborate a solution for the simplified problem without dynamic edge costs and battery constraints. This solution is then extended to the general case in the next subsections.

Energy-optimal path as shortest path. Figure 2 shows a weighted graph, for which we are interested in the shortest path from source s to destination t. First, we consider only the solid lines representing the edges E in the graph. The label of an edge (u, v) denotes its energy costs $c(u, v) = c_L(u, v) + c_P(u, v)$ and is given by two values: the first value $c_L(u, v) \geq 0$ represents the loss of energy and the second $c_P(u, v) = \pi(v) - \pi(u)$ the required energy to overcome the potential energy difference $\pi(v) - \pi(u)$ of u and v. As stated in the previous section, each vertex u in the graph has an elevation $z(u)$ resulting in a potential energy $E_P(z(u)\dots)$, which is abbreviated by $\pi(u)$ and shown next to the label of each vertex. For example, the potential energy $\pi(x)$ is 0, and $\pi(y)$ is 2. Hence, the energy costs $c(x, y)$ of the edge (x, y) is $1 + (2 - 0)$.

Ignoring battery constraints, the problem of finding an energy-optimal path from s to t boils down to finding a shortest path from s to t, which is typically solved using Dijkstra's algorithm (Dijkstra 1959) with worst time complexity $O(n^2)$. However, in our setting it is not applicable, as the potential energy difference of two adjacent vertices can cause a negative weight; e.g. $c(z, t) = 1 + (1 - 4) = -2$. While the Bellman-Ford algorithm (Bellman 1958) works for graphs with arbitrary weights, its worst time complexity of $O(n^3)$ renders this algorithm an inappropriate solution.

A different approach is to first transform the weight function c into a positive reduced weight function c_Π using a so-called *potential function* Π assigning to each vertex a potential, as described in (Mehlhorn and Sanders 2008). In more detail, it is shown that whenever a function Π statisfies $\Pi(v) - \Pi(u) \leq c(u, v)$ and determining c_Π as

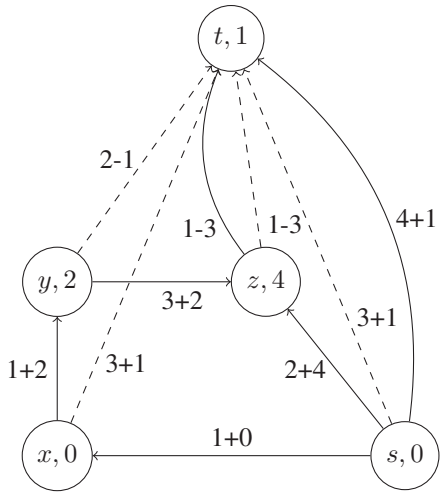

Figure 2: A routing example.

$c_\Pi(u, v) = c(u, v) + \Pi(u) - \Pi(v)$ the shortest paths in (V, E, c_Π) are also shortest paths in (V, E, c) (Mehlhorn and Sanders 2008).

This idea is taken up in Johnson's algorithm by applying the Bellman-Ford algorithm as a preprocessing step to determine the potential function Π, and then solving the shortest path problem in (V, E, c_Π) by Dijkstra's algorithm.

A first, important observation also made in (Neubauer 2010) is that in our specific setting, a potential function Π is naturally obtained without the need of a preprocessing step: the potential energy function π implies a potential function Π, resulting in a positive reduced weight function c_Π.

Lemma 1 *π implies a positive reduced weight function c_π*

Proof.

$$\begin{aligned} c_\pi(u, v) &= c(u, v) + \pi(u) - \pi(v) \\ &= c_L(u, v) + \pi(v) - \pi(u) + \pi(u) - \pi(v) \\ &= c_L(u, v) \geq 0 \end{aligned}$$

□

Therefore the preprocessing step in Johnson's algorithm can be avoided and the problem can be solved in $O(n^2)$.

However, Dijkstra's algorithm has the disadvantage to expand more vertices than necessary. In this example all vertices are expanded before the shortest path (s, z, t) with path cost 4 is found.

Our second contribution to the energy-optimal routing problem is to use A* with a non-trivial heuristic rather than Dijkstra's algorithm.

Algorithm 1 shows a slightly modified version of the A* algorithm (Hart, Nilsson, and Raphael 1968), which determines a shortest path from s to t. It belongs to the informed search algorithms which first pursue paths that appear to be the best routes to t based on the available information. Let $g(v)$ be the current path costs from s to v and $h(v)$ a heuristic estimate for the costs of a shortest path from v to t. The algorithm is initialized by setting all path costs to ∞ except for the source vertex s (line 4). In general $g(s)$ is initialized with

0, and we assume this for the moment as battery constraints are ignored (in subsequent sections, $g(s)$ is initialized with $C - J \geq 0$ following the cost function c_{CJ} developed in the previous section.) The initial vertex s is added to the priority queue Q. In every iteration the vertex u in Q with minimal $g(u) + h(u)$, i.e., for which the current costs plus the estimated costs is minimal, is removed from Q and expanded. During the expansion a successor v of u is added to Q if its new path costs given by $g(u) + c(u, v)$ is smaller than the so-far known value. Note that line 12 is adapted by an additional summand $\hat{c}$; this summand is required in subsequent sections but assumed to be 0 in this example. Moreover, to be able to return the shortest path and not only the minimal weight, the choices for building up a shortes path are recorded via function p in lines 3 and 14.

In (Hart, Nilsson, and Raphael 1968), it is shown that if a heuristic is admissible, i.e. never overestimates the remaining costs to reach t, the algorithm terminates correctly when reaching t. The shortest path $(s, \dots, t)$ from s to t is implicitly given by p and returned in line 9.

Moreover, it was shown that if a heuristic h is *consistent*, which is defined as

$$h(u, t) \leq c(u, v) + h(v, t) \text{ and } h(t, t) = 0 \text{ for all } u, v, t$$

the algorithm terminates correctly in $O(n^2)$ steps. In this case each vertex is expanded at most once. Note that consitency of a heuristic h implies that h is admissible.

Our observation is now that a consistent heuristic is also naturally given in our domain: As mentioned in the previous section, the cost function c_L is based on some physical model of the energy consumption along a road segment $E_L(s, l, \dots)$, where s and l denote the speed and length of the corresponding section. As stated before, it is reasonable to assume that $E_L(s, l, \dots)$ is monotonically increasing in s and linearly increasing in l. Then, we may define a heuristic function h_L in the same way as c_L but taking the minimum $s_{\min}$ over all speed limits and the air line distance between two nodes u and v rather than the possibly longer edge value $l(u, v)$, when it exists. More precisely, let $h_L(u, v) = E_L(s_{\min}, \|(u, v)\|, \dots)$, where $\|(u, v)\|$ denotes the air line distance of u and v. Then

Lemma 2 *Heuristic $h_L(u, t)$ is consistent in (V, E, c_π):*

Proof. Since h_L is linearly increasing in l, $h_L(u, t) \leq h_L(u, v) + h_L(v, t)$. As h_L is monotonic in s and l, $h_L(u, v) \leq c_L(u, v)$, for all u, v, for which $c_L(u, v)$ is defined. Thus, $h_L(u, t) \leq c_L(u, v) + h_L(v, t)$, when $c_L(u, v)$ is defined, showing that h_L is consistent. □

In Figure 2, the respective air line distances are shown as dotted lines.

For example, for the model described in the previous section, we get: Let $s_{\min}$ be the lowest speed limit and $l'(v, t)$ the air line distance from v to t. A lower bound for the energy loss from v to t is then given by $h_L(v, t) = f_r mgl'(v, t) + \frac{1}{2}\rho A c_w s_{\min}^2 l'(v, t)$. Then, the heuristic $h_L(u, t)$ is consistent in (V, E, c_π): Obviously the air line distance l' is a consistent heuristic for the road length l: $l'(u, t) \leq l(u, v) + l'(v, t)$. We write $\nu(u, v) = f_r mg + \frac{1}{2}\rho A c_w s(u, v)^2$ and $\nu' = f_r mg + \frac{1}{2}\rho A c_w s_{\min}^2$ for

Algorithm 1: Energy-A* Algorithm

Input: Directed weighted graph $G = (V, E, c)$ with c representing energy costs, battery capacity C, battery charge J, source vertex s, destination vertex t
Output: A shortest path from s to t

```
1  begin
2      foreach vertex v in V do
3          g(v) ← ∞, p(v) ← null;
4      g(s) ← C − J;
5      Q ← {s};
6      while Q ≠ ∅ do
7          choose u from Q with minimal g(u) + h(u, t);
8          if u = t then
9              return p;
10         Q ← Q \ {u};
11         foreach successor v of u do
12             g′ ← g(u) + c(u, v) + ĉ(u, v, g(u));
13             if g′ < g(v) then
14                 g(v) ← g′, p(v) ← u;
15                 Q ← Q ∪ {v};
```

convenience. Since $s_{\min}$ is a lower bound of $s(u, v)$, also ν' is a lower bound of ν. Hence, consistency follows immediately from

$$\begin{aligned} h_L(u, t) = \nu' l'(u, t) &\leq \nu'(l(u, v) + l'(v, t)) \\ &\leq \nu(u, v) l(u, v) + \nu' l'(v, t) = c_L(u, v) + h_L(v, t) \end{aligned}$$

This approach can be simplified by incorporating the potential function π into the heuristic function h_L to apply the A* algorithm on the original graph (V, E, c). The combined heuristic function h is defined for the vertices u and v by $h(u, v) = h_L(u, v) + h_\pi(u, v)$ where $h_\pi(u, v) = \pi(v) - \pi(u)$ is the potential energy difference between the two vertices.

Lemma 3 *The heuristic h is consistent in (V, E, c).*

Proof. The proof follows immediately from $h_\pi(v, v) = \pi(v) - \pi(v) = 0$ and

$$\begin{aligned} h(u, t) &= h_L(u, t) + h_\pi(u, v) = h_L(u, t) + \pi(t) - \pi(u) \\ &\leq c_L(u, v) + h_L(v, t) + \pi(t) - \pi(u) \\ &= (c_L(u, v) + \pi(v) - \pi(u)) + (h_L(v, t) + \pi(t) - \pi(v)) \\ &= c(u, v) + h(v, t) \end{aligned}$$

□

Theorem 4 *The A* algorithm with heuristic h finds the energy optimal route between two vertices in (V, E, c) with worst time complexity $O(n^2)$.*

Dynamic Weight Calculation. A* is known to be optimally efficient in the sense that it expands the fewest number of vertices among all search algorithms having access to the same heuristics. In addition, a consistent heuristic ensures

that each vertex has to be considered at most once. In our setting, this property is crucial since edge costs in the energy graph may be complex to compute and may depend on a number of varying parameters known only at query time. For example, the mass m of the EV varies with payload, and it is not simply a multiplicative factor in the cost function because it is not part of aerodynamic resistance. As another example, the power demand of auxiliary consumers like the A/C is not known a priori, as it depends on unpredictable variables such as ambient temperature. In fact, the requirement to evaluate the weight function only when necessary and as late in the process as possible rules out most existing preprocessing techniques, such as contraction hierarchies, that are based on global graph analysis.

Battery Constraints. Let us now study the effect of battery constraints for the example in Figure 2. If we assume a fully charged battery at s with $J = C = 5$, the previously shortest path (s, z, t) with path cost 4 is no longer feasible. Although the total costs of the path is below our charge level, it is not possible for the EV to drive the road section (s, z) because it would require 6 energy units. Therefore the energy optimal path is (s, t) with energy costs 5.

Our idea, developed in the following, is to include the battery constraints into the A* algorithm by modifying the weight function c. The modified weight function is then dependent on the path cost function g of the A* algorithm.

In our example, the edge (s, z) with cost 6 cannot be used when $J = C = 5$, and therefore we dynamically increase its cost value to be ∞. Analogously, assume we start in z with $C = 5$ and $J = 4$. If the algorithm expands z, the edge (z, t) offers $c(z, t) = -2$ energy units for recuperation, but the battery can only store $C - J = 5 - 4 = 1$ additional energy unit. Therefore 1 energy unit is lost due to the battery constraint.

These effects can be captured by an additional cost function $\hat{c}$, which is added in line 12 in Algorithm 1:

Definition 2 *For given weight function c and capacity C the function $\hat{c}$ is defined by $\hat{c} : V \times V \times \mathbb{R} \to \mathbb{R}$*

$$(u, v, k) \mapsto \begin{cases} -\Delta(u, v, k) & \text{if } \Delta(u, v, k) < 0 \\ 0 & \text{if } 0 \leq \Delta(u, v, k) \leq C \\ \infty & \text{if } \Delta(u, v, k) > C \end{cases}$$

with $\Delta(u, v, k) := k + c(u, v)$.

This approach, however, extends the A* framework and therefore its optimality and complexity must be verified for the combined cost function $c + \hat{c}$.

First, we show that the heuristic h developed in the previous subsection is still consistent also with respect to $c + \hat{c}$, so that the complexity of Energy-A* remains quadratic:

Lemma 5 *Let h be a consistent heuristic for c and $\hat{c}$ be defined as in Definition 2. Then h is also a consistent heuristic for the cost function $c + \hat{c}$.*

Proof. For $0 \leq \Delta(u, v, k)$ obviously $\hat{c} \geq 0$ holds. In the case of $\Delta(u, v, k) < 0$, $\hat{c}$ equals $-\Delta(u, v, k) \geq 0$. Thus, $\hat{c}$ is non-negative. With consistency of h, we get $h(u, t) \leq c(u, v) + h(v, t) \leq c(u, v) + \hat{c}(u, v, g(u)) + h(v, t)$. □

Next we show that despite the fact that costs are adjusted dynamically, Energy-A* still yields a minimal cost solution:

Lemma 6 *Algorithm Energy-A* returns a minimal-cost solution.*

Proof. Lemma 5 states that every vertex is expanded at most once. Let k be the path cost value $g(u)$ of u when it is expanded. We have to prove that $\hat{c}(u, v, k) = \min\{\hat{c}(u, v, k') | \text{costs } k' \text{ for all possible paths from } s \text{ to } u\}$ holds; i.e. suboptimal paths to u never lead to lower edge weights for outgoing edges of u. This follows because $\hat{c}(u, v, k)$ is monotonically increasing for k. □

Finally, we show that the minimal cost-solution of A* computed in quadratic time adheres to the cost-function c_{CJ} developed in the second section:

Lemma 7 *If a vertex u is expanded in Algorithm 1, the path $P = (v_1, v_2, \ldots, v_k)$ (implicitly given by function p) from $s = v_1$ to $u = v_k$ has path cost $g(u) = c_{CJ}(P)$.*

Proof. We prove $g(u) = c_{CJ}(P)$ by induction on the path length k. The base case $g(s) = C - J = c_{CJ}(s)$ follows immediately from the definition of c_{CJ} and the initialization of the algorithm. Assume $g(v_{k-1}) = c_{CJ}(P^{k-1})$ holds. Then, $g(v_k) = g(v_{k-1}) + c(v_{k-1}, v_k) + \hat{c}(v_{k-1}, v_k, g(v_{k-1})) = c_{CJ}(P^{k-1}) + c(v_{k-1}, v_k) + \hat{c}(v_{k-1}, v_k, c_{CJ}(P^{k-1})) = c_{CJ}(P)$ follows from the inductive step and the definitions of $\hat{c}$ and c_{CJ} because $c(v_{k-1}, v_k) + \hat{c}(v_{k-1}, v_k, c_{CJ}(P^{k-1}))$ equals the cost of the road section (v_{k-1}, v_k) as stated in the definition of path cost function c_{CJ}. □

Altogether, we obtain the following result:

Theorem 8 *The Energy-A* algorithm with heuristic h determines an energy-optimal path in (V, E, c) with worst time complexity $O(n^2)$.*

By using Fibonacci heaps, a time complexity of $O(n \log n + m)$ can be realized. This can be advantageous for real road networks, where the degree of the vertices is typically bounded by some constant and thus the graph is sparse ($m = O(n)$).

Conjecture 9 *The Energy-A* algorithm has complexity $O(n \log n)$ for real road networks.* □

Experimental Results

We developed a prototypic software system for energy efficient routing, based on opensource libraries and freely available data. For a given car type, source address and destination address, the system computes a route with minimum energy costs. The data basis consists of geospatial data from the collaborative OpenStreetMap (OSM) project, and altitude maps of the NASA Shuttle Radar Topographic Mission (SRTM), which provide digital elevation data with a resolution of about 90m. By combining these two sources, we created a road network with elevation and cruising speed information for every point in the network. The costs of road sections corresponding to energy consumption are then calculated dynamically as needed by the A* algorithm.

We compared the performance of our algorithm against two instances of the generic shortest-path framework in

(Artmeier et al. 2010). In this work, a generic algorithmic framework is presented for computing trees of shortest paths starting from a source node s; because trees of paths are considered, path constraints can be added in without affecting optimality of the result. Different instances of the algorithmic framework are obtained by specifying an expansion strategy that determines which vertex to expand next. Two such strategies were considered; the first one (Dijkstra strategy) selects a vertex with the lowest distance from the source vertex. This has an exponential worst time complexity in the presence of negative weights, but was shown to be one of the best strategies in (Artmeier et al. 2010). The second instance (Pallottino strategy) implements the algorithm in (Pallottino 1984), a variant of the Bellman-Ford algorithm with a worst time complexity $O(n^2m)$ but with a favorable performance (Zhan and Noon 1998) in real world road networks.

We evaluated these three algorithms on a section of the OSM map representing Bavaria, a state of Germany. This road graph contains 2,423,313 vertices and 4,983,944 edges. The parameters of our specific electric vehicle model used in these experiments were $c_w = 0.42$, $\eta_r = \eta_c = 0.8$, $A = 2.0$ m^2, $m = 1000$ kg and $C = 25$ kWh. This charge allows a cruising range of around 150 kilometers. The randomly selected sampling set of source and targets nodes was divided into 10 sample classes with different air line distances. Each of the 10 classes consists of $k = 100$ samples. The experiments were carried out on a Intel Core2 Duo CPU with 2.20 GHz and 2 GB RAM. Table 1 shows the mean runtime ($\pm$ standard deviation) in seconds for computing energy-optimal paths of each sample class. Memory consumption of the Energy-A* algorithm was found to increase only roughly linearly with distance.

The results indicate that our Energy-A* algorithm is faster than the generic framework with the Dijkstra or Pallottino strategy. Especially for small distances between source and destination vertex, the improvement is significant. In addition, our algorithm also leads to fewer node expansions and thus fewer (potentially expensive) evaluations of the energy cost function.

Distance (air line)	Energy-A*	Dijkstra strategy	Pallottino strategy
[0,10]	0.03 ± 0.01	1.55 ± 0.43	4.96 ± 7.52
]10,20]	0.04 ± 0.01	1.60 ± 0.41	5.10 ± 7.29
]20,30]	0.07 ± 0.02	1.55 ± 0.43	5.92 ± 11.44
]30,40]	0.11 ± 0.04	1.62 ± 0.42	5.40 ± 7.97
]40,50]	0.17 ± 0.05	1.59 ± 0.41	8.11 ± 12.88
]50,60]	0.24 ± 0.07	1.70 ± 0.41	9.99 ± 16.53
]60,70]	0.31 ± 0.09	1.66 ± 0.49	7.11 ± 11.59
]70,80]	0.41 ± 0.10	1.62 ± 0.41	6.75 ± 15.71
]80,90]	0.52 ± 0.14	1.62 ± 0.40	7.01 ± 12.51
]90,100]	0.60 ± 0.16	1.65 ± 0.42	5.46 ± 7.98

Table 1: Mean runtimes ($\pm$ standard deviation) in seconds for computing energy-optimal paths.

Conclusion

Optimal routing for electric vehicles with rechargeable batteries will become increasingly important in the future. We formalized this problem in a graph-theoretic context as an instance of a shortest path problem with additional path-related costs. We then showed how specific properties of the energy domain can be exploited to obtain a fast $O(n^2)$ routing algorithm that can handle dynamic and path-related costs and improves the so far known complexity bound.

It is easy to modify our algorithm to perform energetic reachability analysis and therefore prediction of the remaining range of vehicles. Further research will also study the impact of the negative/positive edge ratio in graphs and refinements to our vehicle model, in particular including kinetic energy. In addition, we plan to extend our approach to stochastic models in order to be able to calculate the risk of running out of energy before arriving at the destination.

References

Artmeier, A.; Haselmayr, J.; Leucker, M.; and Sachenbacher, M. 2010. The shortest path problem revisited: Optimal routing for electric vehicles. In *KI'10*.

Bast, H.; Funke, S.; Sanders, P.; and Schultes, D. 2007. Fast routing in road networks with transit nodes. *Science* 316(5824):566.

Bellman, R. 1958. On a routing problem. *Quarterly of Applied Mathematics.* 16(1):87–90.

Dijkstra, E. 1959. A note on two problems in connexion with graphs. *Numerische Mathematik* 1(1):269–271.

Garey, M., and Johnson, D. 1979. *Computers and Intractibility: A Guide to the Theory of NP-Completeness*. W. H. Freeman, New York.

Geisberger, R.; Sanders, P.; Schultes, D.; and Delling, D. 2008. Contraction hierarchies: Faster and simpler hierarchical routing in road networks. In *Proc. WEA'08*.

Hart, P.; Nilsson, N.; and Raphael, B. 1968. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Trans. on Sys. Science and Cybernetics* 4(2):100–107.

Johnson, D. B. 1977. Efficient algorithms for shortest paths in sparse networks. *Journal of the ACM* 24(1):1–13.

Joksch, H. C. 1966. The shortest route problem with constraints. *J. of Math. Analysis and Applications* 14:191–197.

Mehlhorn, K., and Sanders, P. 2008. *Data Structures and Algorithms. The Basic Toolbox*. Springer.

Neubauer, S. 2010. Planung energieeffizienter Routen in Straßennetzwerken. Master's thesis, Karlsruhe Inst. of Tech.

Pallottino, S. 1984. Shortest-path methods: Complexity, interrelations and new propositions. *Networks* 14(2):257–267.

Sanders, P., and Schultes, D. 2005. Highway hierarchies hasten exact shortest path queries. In *ESA'05*.

Zhan, F. B., and Noon, C. E. 1998. Shortest path algorithms: An evaluation using real road networks. *Transportation Science* 32:65–73.