Minimum-cost paths for electric cars

Dani Dorfman[*]   Haim Kaplan[*]   Robert E. Tarjan[†]   Mikkel Thorup[‡]   Uri Zwick[*]

Abstract

An electric car equipped with a battery of a finite capacity travels on a road network with an infrastructure of charging stations. Each charging station has a possibly different cost per unit of energy. Traversing a given road segment requires a specified amount of energy that may be positive, zero or negative. The car can only traverse a road segment if it has enough charge to do so (the charge cannot drop below zero), and it cannot charge its battery beyond its capacity. To travel from one point to another the car needs to choose a *travel plan* consisting of a path in the network and a recharging schedule that specifies how much energy to charge at each charging station on the path, making sure of having enough energy to reach the next charging station or the destination. The cost of the plan is the total charging cost along the chosen path. We reduce the problem of computing plans between every two junctions of the network to two problems: Finding optimal energetic paths when no charging is allowed and finding standard shortest paths. When there are no negative cycles in the network, we obtain an $O(n^3)$-time algorithm for computing all-pairs travel plans, where n is the number of junctions in the network. We obtain slightly faster algorithms under some further assumptions. We also consider the case in which a bound is placed on the number of rechargings allowed.

1 Introduction

An electric car with a battery that can store up to B units of energy travels in a road network with an infrastructure of charging stations. The road network is represented by a weighted directed graph $G = (V, A, c, r)$, where V is the set of vertices (junctions), $A \subseteq V \times V$ is the set of arcs (road segments), $c : A \to \mathbb{R}$ and $r : V \to \mathbb{R}^+ \cup \{\infty\}$. We always let $n = |V|$ and $m = |A|$. For $a \in A$, $c(a)$ is the amount of energy needed to traverse arc a. This amount can be positive, zero or negative. A negative cost may indicate a downhill segment that can be used to charge the battery. The battery can be charged, at a certain cost, at some vertices of the network. For every $v \in V$, $r(v) \geq 0$ is the price for one unit of energy at v. If $r(v) < \infty$, we say that v is a *charging station*. If $r(v) = \infty$ then the battery cannot be charged at v. The car can traverse an arc $(u, v) \in A$ only if it is in u and the charge $b \geq 0$ in its battery satisfies $b \geq c(u, v)$. It then reaches v with a charge of $\min\{b - c(u, v), B\}$ in its battery. The charge b of the battery always satisfies $0 \leq b \leq B$, i.e., the charge in the battery cannot drop below zero and cannot exceed the capacity B.

If $c : A \to \mathbb{R}^+$, i.e., all arc costs are non-negative, then the graph $G = (V, A, c, r)$ models a traditional road network with an infrastructure of gas stations in which fuel-based cars can travel.

To travel from s to t the car needs to choose a *travel plan* composed of a directed path P from s to t, and a *recharging schedule* that specifies how much energy to charge at every charging station on P, making sure that the car always has enough energy to continue its journey. We assume that the car starts at s with an empty battery. The *cost* of the plan is the total charging cost along the selected path. We let $\rho_B(s, t)$ be the minimum cost of a travel plan from s to t, starting at s with an empty battery, where B is the capacity of the battery. We also let $\rho_{B, \Delta}(s, t)$ be the minimum cost of a plan that uses at most Δ charging stops.

Khuller, Malekian and Mestre [9] considered the problem of finding plans in the context of conventional fuel-based cars, i.e., $c : A \to \mathbb{R}^+$, and obtained an $O(n^3 \min\{\Delta \log n, \Delta^2\})$-time algorithm for the all-pairs version,

where $\Delta \leq n$ is a bound on the number of chargings allowed. When no bound is placed on the number of times the battery can be charged, the running time of their algorithm is $O(n^4 \log n)$. They also obtained an $O(\min\{n^3, \Delta n^2 \log n\})$-time algorithm for the single-target version of the problem.

Dorfman et al. [4] recently considered a related problem of finding optimal paths for electric cars when no rechargings along the way are allowed, but when arc costs are also allowed to be negative. When there are no negative arc costs, the problem is equivalent to the standard shortest paths problem. When there are negative arc costs the problem becomes a strict extension of the standard shortest paths problem. The problem is also essentially a special case of the problem considered in this paper.

In this paper we present algorithms for the all-pairs version of the minimum-cost plans problem, allowing rechargings along the way. The network may contain negative arc costs. When there are no negative cycles, or when the capacity B of the battery is sufficiently large, we obtain an $O(n^3)$-time algorithm. This improves and generalizes the $O(n^4 \log n)$-time algorithm of Khuller et al. [9], which does not allow negative arc costs. Our $O(n^3)$ time bound also applies when at most $\Delta \leq n$ rechargings are allowed, improving the $O(n^3 \min\{\Delta \log n, \Delta^2\})$-time bound of Khuller et al. [9].

We obtain our new results by showing that finding plans that allow rechargings along the way can be reduced in a simple way to the problem studied by Dorfman et al. [4], i.e., the problem of finding non-recharging optimal paths. More specifically, the optimal energetic costs found by the algorithms of Dorfman et al. [4] are used to set up a constant number of min-plus products. The results of these min-plus products are used as arc costs in an auxiliary graph on which a standard APSP problem needs to be solved.

The rest of the paper is organized as follows. In the next section we review the results of Dorfman et al. [4] for the no-rechargings case. In Section 3 we show that the problem of finding plans can be reduced to finding optimal paths when no rechargings are allowed and to computing standard shortest paths. In Section 4 we use the reduction to obtain our efficient algorithms for finding plans. In Section 5 we describe a simple reduction in the opposite direction, from computing standard shortest paths to computing plans, showing that the algorithms obtained for finding plans are essentially optimal. In Section 6 we consider settings in which the initial charge of the car is not zero. We end in Section 7 with some concluding remarks and open problems.

2 No rechargings allowed - the quantities $\alpha_{B,a}(s,t)$ and $\beta_{B,b}(s,t)$

We review the no-rechargings results of Dorfman et al. [4] and introduce the quantities $\alpha_{B,a}(s,t)$ and $\beta_{B,b}(s,t)$ that play an important role in the reduction presented in the next section.

Let $\alpha_{B,a}(s,t)$ be the Maximum Final Charge (MFC) with which the car can reach t if it starts in s with charge a, where $0 \leq a \leq B$, in its battery, when *no* recharging is allowed along the way. If it is not possible to get from s to t with an initial charge of a at s, we let $\alpha_{B,a}(s,t) = -\infty$. We will mostly be interested in the cases $a = 0$ or $a = B$, i.e., starting from s with either an empty or a full battery.

Let $\beta_{B,b}(s,t)$ be the Minimum Initial Charge (MIC) needed in s in order to reach t with a charge of at least b when again *no* recharging is allowed along the way. If it is not possible to get to t with a charge of b even if we start from s with a charge of B, i.e., with a full battery, we let $\beta_{B,b}(s,t) = \infty$. Again, we are mostly interested in the cases $b = 0$ or $b = B$.

If all arc costs are non-negative then it is easy to see that $\alpha_{B,a}(s,t) = a - \delta(s,t)$, if $a \geq \delta(s,t)$, and $\alpha_{B,a}(s,t) = -\infty$ otherwise, where $\delta(s,t)$ is the standard distance from s to t in the graph, i.e., the length of the shortest path from s to t with respect to the cost function c. Similarly, $\beta_{B,b}(s,t) = \delta(s,t) + b$, if $\delta(s,t) + b \leq B$, and $\beta_{B,b}(s,t) = \infty$ otherwise. Thus, if all arc costs are non-negative, then $\alpha_{B,a}(s,t)$ and $\beta_{B,b}(s,t)$ for all $s,t \in V$ and $a, b \in \{0, B\}$, can be easily computed after solving a standard All-Pairs Shortest Paths (APSP) problem.

When arc costs can be both positive and negative, computing all values of $\alpha_{B,a}(s,t)$ and $\beta_{B,b}(s,t)$ is more difficult. Dorfman et al. [4] gave an $O(mn + n^2 \log n)$-time algorithm for the all-pairs versions when there can be negative arcs but there are no negative cycles and an $O(mn^2 + n^3 \log n)$-time algorithm when there can be negative cycles. They also gave an $O(mn + n^2 \log n)$-time algorithm for the all-pairs versions when there can be negative cycles, if the capacity of the battery is sufficiently large, namely $B \geq 3nM$, where $M = \max_{(u,v) \in A} |c(u,v)|$. [1]

The algorithms of Dorfman et al. [4], which we do not repeat here, work with another quantity $\delta_{B,a}(s,t) = a - \alpha_{B,a}(s,t)$, the *minimum depletion* of the battery when starting at s with a charge a in the battery. This quantity is closely related to the standard distance $\delta(s,t)$ from s to t in the graph and in some cases equal to it.

---

[1]This can be improved to $B \geq nM$.

They show that $\delta_{B,a}(s,t)$ can be computed by suitable adaptations of the standard Bellman-Ford and Dijkstra algorithms. They also show that $\beta^G_{B,b}(s,t) = B - \alpha^{\overleftarrow{G}}_{B,B-b}(t,s)$, where $\overleftarrow{G}$ is the graph obtained from G by reversing the direction of all arcs, maintaining the arc costs. Thus, algorithms for computing $\alpha_{B,a}(s,t)$ can also be used to compute $\beta_{B,b}(s,t)$, and vice versa. (This is true for the all-pairs version of the problems. A single-source algorithm for the maximum final charges $\alpha_{B,a}(s,t)$ becomes a single-target algorithm for the minimum initial charges $\beta_{B,b}(s,t)$.)

3 Handling rechargings by reduction to no rechargings

In this section we describe a simple reduction from computing $\rho_B(s,t)$, for every $s,t \in V$, to the computation of $\alpha_{B,a}(s,t)$ and $\beta_{B,b}(s,t)$, for every $s,t \in V$ and $a,b \in \{0,B\}$, and to the solution of a standard all-pairs shortest paths (APSP) problem on an auxiliary network.

No assumptions are made in this section on $c : A \to \mathbb{R}$. Arc costs may be negative and the graph may even contain negative cycles. ($\rho_B(s,t)$ is well-defined even in the presence of negative cycles.) For simplicity we assume that all finite charging costs $r(v)$ are distinct. (Ties can be broken easily.)

3.1 Travel plans We begin with a formal definition of *travel plans*. We use u^a, where $u \in V$ and $a \in [0,B]$ to denote the state in which the car is at u with a charge of a in its battery. A travel plan is then simply a sequence of states such that the move between two consecutive states corresponds to either traversing an arc of the graph, or recharging the battery by a certain amount.

DEFINITION 3.1. (TRAVEL PLANS) *A travel plan P from s to t in a graph $G = (V,A,c,r)$ is a sequence $u_0^{a_0} u_1^{a_1} \ldots u_\ell^{a_\ell}$, where $s = u_0, u_1, \ldots, u_\ell = t \in V$, $0 = a_0$, $0 \le a_i \le B$ for all $0 \le i \le \ell$, and such that for every $i = 0, 1, \ldots, \ell - 1$, either $(u_i, u_{i+1}) \in A$, $a_i \ge c(u_i, u_{i+1})$ and $a_{i+1} = \min\{a_i - c(u_i, u_{i+1}), B\}$, corresponding to legally traversing an arc, or $u_i = u_{i+1}$ and $a_i < a_{i+1}$, corresponding to a charging of the battery. The cost of the plan is $cost(P) = \sum_{i \in R} r(u_i)(a_{i+1} - a_i)$, where $R = \{0 \le i \le \ell \mid u_i = u_{i+1}\}$ is the set of indices in which rechargings take place.*

If $P = u_0^{a_0} u_1^{a_1} \ldots u_\ell^{a_\ell}$ is a travel plan, then $u_0, u_1, \ldots, u_\ell$ is the path used by the plan, where each vertex in which a recharging takes place has two consecutive appearances. A minimum-cost plan from s to t is of course a plan from s to t whose cost is minimum. If the graph contains no negative cycles, it is not difficult to see that a path used by an optimal plan may be assumed to be simple. This is not true, however, in the presence of negative cycles.

Conceptually, we can define an infinite graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \ell)$, where $\mathcal{V} = \{u^a \mid u \in V, a \in [0,B]\}$, $\mathcal{E} = \mathcal{E}_1 \cup \mathcal{E}_2$, $\mathcal{E}_1 = \{(u^a, v^b) \mid (u,v) \in A, a \ge c(u,v), b = \min\{a - c(u,v), B\}\}$, $\mathcal{E}_2 = \{(u^a, u^b) \mid u \in V, a < b\}$, and where $\ell(u^a, v^b) = 0$, for every $(u^a, v^b) \in \mathcal{E}_1$, and $\ell(u^a, u^b) = r(u)(b-a)$, for every $(u^a, u^b) \in \mathcal{E}_2$. Then a travel plan is simply a path in $\mathcal{G}$, and the cost of the plan is simply the length of the path with respect to the length function ℓ. The challenge, of course, is to efficiently compute shortest paths in this implicit infinite graph. (If all arc costs are integral, the graph becomes finite, but its size is $O(B(m+n))$ which is still too large.)

3.2 Structure of optimal plans Each plan P from s to t can be partitioned into segments $P = P_0 P_1 \ldots P_k$ such that no rechargings take place within each segment and such that a recharging does take place when moving from segment to segment. Each segment P_i is then of the form $P_i = x_i^{b_i} \cdots x_{i+1}^{c_i}$, where $c_i < b_{i+1}$, if $i < k$. Also $s = x_0$ and $t = x_{k+1}$. The whole plan P is then $x_0^{b_0} \cdots x_1^{c_0} | x_1^{b_1} \cdots x_2^{c_1} | x_2^{b_2} \cdots x_3^{c_2} | \cdots | x_{k-1}^{b_{k-1}} \cdots x_k^{c_{k-1}} | x_k^{b_k} \cdots x_{k+1}^{c_k}$, where each vertical line separates two segments and indicates a place at which a charging takes place. The vertices $x_1, x_2, \ldots, x_k$ are the vertices in which rechargings take place. (If no rechargings take place along P, then $k = 0$ and $P = P_0$. If a charging takes place at s, then $s = x_0 = x_1$ and $P_0 = s^0$.)

The following two lemmas are extensions of similar lemmas of Khuller, Malekian and Mestre [9].

LEMMA 3.1. *Let P be a minimum-cost plan from s to t. Let $P = P_0 P_1 \ldots P_k$ a partition of P into segments as above, and let $P_i = x_i^{b_i} \cdots x_{i+1}^{c_i}$, where $c_i < b_{i+1}$, if $i < k$. Recall that $x_1, x_2, \ldots, x_k$ are the vertices on P in which rechargings take place and that $s = x_0$ and $t = x_{k+1}$. Then, for $i = 1, \ldots, k-1$,*

(i) *If $r(x_i) > r(x_{i+1})$, then P_i contains a state y_i^0, i.e., a vertex y_i at which the car has an empty battery. (The vertex y_i may be x_{i+1} but not x_i.)*
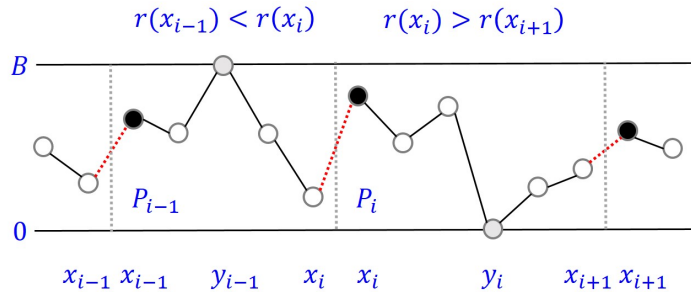
Figure 1: A schematic description of Lemmas 3.1 and 3.2. The height of a vertex represents the charge in the battery. Dotted red lines represent rechargings.

(ii) *If $r(x_i) < r(x_{i+1})$, then P_i contains a state y_i^B, i.e., a vertex y_i at which the car has a full battery. (The vertex y_i may be x_i but not x_{i+1}.)*

Proof. Let $1 \le i < k$ and let $P_i = u_1^{a_1} u_2^{a_2} \ldots u_\ell^{a_\ell}$, where $u_1^{a_1} = x_i^{b_i}$ and $u_\ell^{a_\ell} = x_{i+1}^{c_i}$. Let $d_i = b_i - c_{i-1} > 0$ and $d_{i+1} = b_{i+1} - c_i > 0$ be the amount of charge added to the battery at x_i and x_{i+1}, respectively.

(i) Assume that $r(x_i) > r(x_{i+1})$. Let $\varepsilon = \min\{a_1, a_2, \ldots, a_\ell\}$. If $\varepsilon = 0$, we are done, as there is an index $1 \le j \le \ell$ such that $a_j = 0$. (Note that $j \ge 2$ as $a_1 > 0$.) Otherwise, let $\delta = \min\{d_i, \varepsilon\} > 0$. We can then replace the segment P_i in P with the segment $P_i' = u_1^{a_1-\delta} u_2^{a_2-\delta} \ldots u_\ell^{a_\ell-\delta}$. This is equivalent to charging δ less at x_i and δ more at x_{i+1}. The resulting plan P' is valid and $cost(P') = cost(P) - \delta\,(r(x_i) - r(x_{i+1})) < cost(P)$, contradicting the optimality of P.

(ii) Assume that $r(x_i) < r(x_{i+1})$. Let $\varepsilon = B - \max\{a_1, a_2, \ldots, a_\ell\}$. If $\varepsilon = 0$, we are done, as there is an index $1 \le j \le \ell$ such that $a_j = B$. (Note that $j < \ell$ as $a_\ell < B$.) Otherwise, let $\delta = \min\{d_{i+1}, \varepsilon\} > 0$. We can then replace the segment P_i in P with the segment $P_i' = u_1^{a_1+\delta} u_2^{a_2+\delta} \ldots u_\ell^{a_\ell+\delta}$. This is equivalent to charging δ more at x_i and δ less at x_{i+1}. The resulting plan P' is valid and $cost(P') = cost(P) + \delta\,(r(x_i) - r(x_{i+1})) < cost(P)$, again contradicting the optimality of P. $\square$

See Figure 1 for a schematic description of Lemma 3.1 and the following Lemma 3.2 which follows almost immediately from Lemma 3.1.

LEMMA 3.2. *Let P be a minimum-cost plan from s to t. Then, there are states $y_0^{a_0}, y_1^{a_1} \ldots, y_k^{a_k}$ on P such that $s = y_0$, $t = y_k$, $a_0 = 0$, and $a_i \in \{0, B\}$, for $i = 1, \ldots, k-1$, and such that at most one recharging takes place between $y_i^{a_i}$ and $y_{i+1}^{a_{i+1}}$ on P, for $i = 0, \ldots, k-1$. (In other words y_i is either reached with an empty battery, or left with a full battery, for each $i = 0, 1, \ldots, k-1$. Note that a_k, the final charge at $y_k = t$, is not required to be in $\{0, B\}$.)*

Proof. Let $P = P_0 P_1 \ldots P_k$ be the partition of P as in Lemma 3.1. By the lemma, each segment P_i, for $i = 1, 2, \ldots, k-1$, contains a state $y_i^{a_i}$, where $a_i \in \{0, B\}$. Let $y_0^{a_0} = s^0$ and $y_k^{a_k} = t^{a_k}$, where a_k is the final charge at t according to P. The sequence $y_0^{a_0}, y_1^{a_1} \ldots, y_k^{a_k}$ satisfies all the required properties. In fact, there is exactly one recharging between $y_i^{a_i}$ and $y_{i+1}^{a_{i+1}}$ on P, for $i = 0, \ldots, k-1$, unless $k = 0$, in which case $y_0^{a_0} = s^0$, $y_1 = t^{a_1}$ and there is no recharging between them. $\square$

We note that in the sequence $y_0^{a_0}, y_1^{a_1} \ldots, y_k^{a_k}$ whose existence is proved in Lemma 3.2 we may have $y_i = y_{i+1}$, in which case $a_i = 0$ and $a_{i+1} = B$. This can happen if $y_i^{a_i}$ is the last state in P_i and $y_{i+1}^{a_{i+1}}$ is the first state in P_{i+1}.

If there are no negative arc costs, then a minimum-cost plan from s to t always reaches t with an empty battery. This is not necessarily the case when arc costs may be negative.

3.3 The reduction Lemma 3.2 suggests the following algorithm for computing $\rho_B(s, t)$ for every $s, t \in V$. Construct a complete graph $G^{0,B}$ on the vertex set $V^{0,B} = V^0 \cup V^B$, where $V^a = \{u^a \mid u \in V\}$, for $a \in \{0, B\}$. Here, u^a corresponds to being at u with charge of at least a in the battery. We define a new cost function ℓ_B such that $\ell_B(u^a, v^b)$, where $u, v \in V$ and $a, b \in \{0, B\}$, is the minimum cost of getting from u to v, starting with an

initial charge of a at u and reaching v with a charge of at least b, when *at most one recharging* is allowed along the way. By Lemma 3.2, computing minimum-cost plans in the original graph would then reduce to finding shortest paths in $G^{0,B}$ with respect to the cost function ℓ_B. More specifically, $\rho_B(s,t) = \delta_{\ell_B}(s^0, t^0)$, for every $s, t \in V$, where $\delta_{\ell_B}(s^0, t^0)$ is the distance from s^0 to t^0 with respect to ℓ_B. (Note that t^0 here stands for reaching t with a charge of at least 0. When there are negative arc costs, a minimum-cost plan may reach the destination with strictly positive charge.)

We next describe how the new costs $\ell_B(u^a, v^b)$ are computed. This is the main step of our reduction. Consider the cheapest way of getting from u^a to v^b, i.e., from u with initial charge a to v with final charge at least b, with (an optional) recharging at a given vertex $x \in V$ along the way. Recharging is not allowed in any other vertex. (We do not assume that u, v and x are distinct. We may even have $u = v = x$.) Clearly, we should choose a path from u^a to x that reaches x with the maximum possible charge, namely $\alpha_{B,a}(u, x)$. We then buy just enough charge at x to reach v with a charge of at least b. The minimum initial charge at x required to reach v^b is $\beta_{B,b}(x, v)$. The amount of charge we need to buy at x is thus $(\beta_{B,b}(x, v) - \alpha_{B,a}(u, x))^+$, where $z^+ = \max\{0, z\}$. (If $\alpha_{B,a}(u, x) > \beta_{B,b}(x, v)$, we do not need to recharge at x.) Considering all choices for the recharging vertex x, including u and v, we get:

$$\ell(u^a, v^b) = \min_{x \in V} r(x) \cdot (\beta_{B,b}(x, v) - \alpha_{B,a}(u, x))^+ .$$

We allow the case $x = u$, in which case $\alpha_{B,a}(u, u) \geq a$. (Strict inequality is possible in the presence of negative cycles.) We also allow the case $x = v$, in which case $\beta_{B,b}(v, v) \leq b$. (Again a strict inequality is possible in the presence of negative cycles.)

To compute $\ell(u^a, v^b)$ we first remove the $+$ from the definition of $\ell(u^a, v^b)$, namely,

$$\ell'(u^a, v^b) = \min_{x \in V} r(x) \cdot (\beta_{B,b}(x, v) - \alpha_{B,a}(u, x)) .$$

It is then easy to see that

$$\ell(u^a, v^b) = \ell'(u^a, v^b)^+ ,$$

as if $\ell'(u^a, v^b) < 0$ then it is possible to get from u^a to v^b without recharging, i.e., at cost 0.

To compute $\ell'(u^a, v^b)$ we form a layered graph $(V^{0,B} \times U) \cup (U \times V^{0,B})$, where $U = \{x \in V \mid r(x) < \infty\}$. (We assume that different copies of $V^{0,B}$ are used in the first and third layers of this graph, see Figure 2.) We let

$$w(u^a, x) = -r(x)\alpha_{B,a}(u, x) \qquad , \qquad w(x, v^b) = r(x)\beta_{B,b}(x, v) .$$

Now $\ell'(u^a, v^b)$ can be computed using a single min-plus product.[2] The time required is $O(n^3)$ using the naïve algorithm, or slightly faster using the algorithm of Williams [13, 14], assuming that all $\alpha_{B,a}(u, x)$ and $\beta_{B,b}(x, v)$ values are given to us.

The following lemma formally proves the correctness of the reduction. Recall that $\delta_{\ell_B}(s^0, t^0)$ is the distance from s^0 to t^0 in the graph $G^{0,B}$ with respect to ℓ_B.

LEMMA 3.3. $\rho_B(s,t) = \delta_{\ell_B}(s^0, t^0)$, *for every* $s, t \in V$.

Proof. Any path P from s^0 to t^0 in the complete graph $G^{0,B}$ with the cost function ℓ_B corresponds to a plan P' from s to t in the original graph G whose cost is exactly $\ell_B(P)$. More specifically, let P be the path $s = y_0^0, y_1^{a_1}, \ldots, y_{k-1}^{a_{k-1}}, y_k^0 = t$. Then each arc $(y_i^{a_i}, y_{i+1}^{a_{i+1}})$ can be replaced by a plan in the original graph G whose cost is $\ell(y_i^{a_i}, y_{i+1}^{a_{i+1}})$. Concatenating all these plans we get a plan P' in G whose total cost is $\ell_B(P) = \sum_{i=0}^{k-1} \ell(y_i^{a_i}, y_{i+1}^{a_{i+1}})$. Thus, $\rho_B(s,t) \leq \delta_{\ell_B}(s^0, t^0)$.

Conversely, let P' be a minimum-cost plan from s to t in the original graph G. By Lemma 3.2 there are states $y_0^{a_0}, y_1^{a_1}, \ldots, y_k^{a_k}$ on P such that $y_0^{a_0} = s^0$, $y_k = t$, $a_0, \ldots, a_{k-1} \in \{0, B\}$, and such that at most one recharging takes place between $y_i^{a_i}$ and $y_{i+1}^{a_{i+1}}$. Let P be the path $s = y_0^0, y_1^{a_1}, y_2^{a_2} \ldots, y_k^0$ in $G^{0,B}$. (Note that we replaced a_k here by 0, possibly throwing away extra charge with which the plan reaches t.) The cost $\ell_B(P)$ of P is at most the cost of the plan P'. Thus, $\rho_B(s,t) \geq \delta_{\ell_B}(s^0, t^0)$. $\square$

---

[2]The min-plus product of an $n \times p$ matrix $A = (a_{i,j})$ and a $p \times n$ matrix $A = (a_{i,j})$, is the $n \times n$ matrix $C = (c_{i,j})$ defined by $c_{i,j} = \min_{k=1}^p a_{i,k} + b_{k,j}$, for every $1 \leq i, j \leq n$.
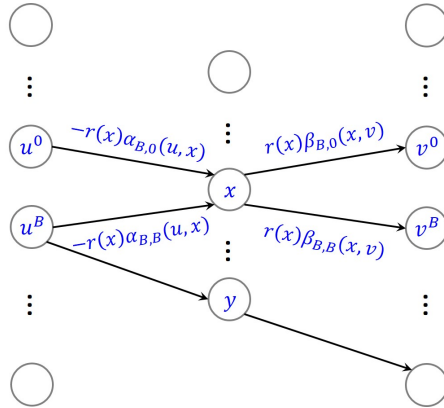
Figure 2: The min-plus product used to compute $\ell'(u^a, v^b)$, for every $u, v \in V$ and $a, b \in \{0, B\}$. The vertices in the middle layer are the vertices $x \in V$ for which $r(x) < \infty$.

Let $MCP(m, n)$ denote the the problem of computing all-pairs *Minimum-Cost Plans* on a graph with n vertices and m arcs, i.e., the computation of $\rho_B(s, t)$ for every $s, t \in V$. Let $MFC(m, n)$ denote the problem of computing all-pairs *Maximum Final Charges*, i.e., the computation of $\alpha_{B,a}(s, t)$ for every $s, t \in V$, for a given $0 \le a \le B$, on a graph with n vertices and m arcs. Let $MIC(m, n)$ denote the problem of computing all-pairs *Minimum Initial Charges*, i.e., the computation of $\beta_{B,b}(s, t)$ for every $s, t \in V$, for a given $0 \le b \le B$, again on a graph with n vertices and m arcs. Let $MPP(n, p)$ be the problem of computing a *Min-Plus Product* of an $n \times p$ matrix and a $p \times n$ matrix. Finally, let $APSP(m, n)$ be the problem of computing the All-Pairs Shortest Paths, i.e., $\delta(s, t)$ for every $s, t \in V$, on a graph with n vertices and m edges.

Putting everything together, we get:

THEOREM 3.1. *$MCP(m, n)$ can be reduced to two instances of $MFC(m, n)$ and $MIC(m, n)$ each, an instance of $MPP(2n, p)$, where $p \le n$ is the number of charging stations, and an instance of $APSP(4n^2, 2n)$.*

We need two instances of $MFC(m, n)$, for $a = 0, B$, and two instances of $MIC(m, n)$, for $b = 0, B$. As noted, $MIC(m, n)$ can be reduced to $MFC(m, n)$ by simply reversing the graph. It is also interesting to note that $APSP(n^2, n)$ can be reduced to a collection of $MPP(n, n)$ problems that can be solved in the time needed for a single $MPP(n, n)$ instance (see Aho et al. [1]). Also, $MPP(n, p)$ can easily be reduced to $APSP(2np, 2n + p)$.

Similarly, if we let $MCP_\Delta(m, n)$ be the problem of computing $\rho_{B,\Delta}(s, t)$ for every $s, t \in V$, i.e., computing minimum-cost plans between every pair of vertices when at most Δ rechargings are allowed, and $APSP_\Delta(u, v)$ be the problem of computing all-pairs shortest paths that are allowed to contain at most Δ arcs, then we have

THEOREM 3.2. *$MCP_\Delta(m, n)$ can be reduced to two instances of $MFC(m, n)$ and $MIC(m, n)$ each, an instance of $MPP(2n, p)$, where $p \le n$ is the number of charging stations, and an instance of $APSP_\Delta(4n^2, 2n)$.*

4 Algorithms for minimum-cost plans

Using the reduction of the previous section and the best available algorithms for the Maximum Final Charge (*MFC*), Minimum Initial Charge (*MIC*), Min-plus product (*MPP*) and All-Pairs Shortest Paths (*APSP*) we obtain the following results:

THEOREM 4.1. *The all-pairs version of the Minimum-Cost Plans (MCP) problem in a graph with no negative cycles can be solved in $O(\frac{n^3}{2^{c\sqrt{\log n}}} + mn) = O(n^3)$ time, for some $c > 0$. The same time bound, with a different constant c, applies when at most $\Delta \le n$ rechargings can be used on each path.*

Proof. Dorfman et al. [4] gave algorithms for solving the *MFC* and the *MIC* problems in $O(mn + n^2 \log n)$ time, on graphs with no negative cycles. The *MPP* and *APSP* problems can be solved in $O(\frac{n^3}{2^{c\sqrt{\log n}}})$ time using a randomized algorithm of Williams [13, 14], or a deterministic algorithm of Chan and Williams [2], for some $c > 0$.

When a bound Δ is placed on the number of chargings, we need to solve an $APSP_\Delta(n^2, n)$ problem, rather than an $APSP(n^2, n)$ problem. It is well-known that such a problem can be solved by computing $O(\log \Delta)$ MPP problems, [3] and hence can also be solved in $O(\frac{n^3}{2^{c\sqrt{\log n}}})$ time, for some $c > 0$. (Note that $\frac{n^3 \log n}{2^{c\sqrt{\log n}}} = O(\frac{n^3}{2^{c'\sqrt{\log n}}})$ for any $c' < c$.) $\quad \square$

The bound in Theorem 4.1 also applies when there may be negative cycles, if the capacity B of the battery is sufficiently large, i.e., $B \geq nM$, where $M = \max_{(u,v) \in A} |c(u,v)|$. (For simplicity, we did not include this in the statement of the theorem.)

We note that the instance of the $APSP$ problem that we need to solve is dense, i.e., $m = n^2$. Thus, efficient algorithms for sparse instances of the $APSP$ problem, such as the $O(mn + n^2 \log n)$-time algorithm obtained by running Dikstra's [3] algorithm from every vertex, implemented using Fibonacci heaps [7] or Hollow heaps [8], or the slightly faster $O(mn + n^2 \log \log n)$-time algorithm of Pettie [10], would both require $O(n^3)$ time. We also note that a slightly faster algorithm for solving the $APSP_\Delta(n^2, n)$ problem can be obtained using the sampling technique of Zwick [16], showing that when $\Delta > \log n$, the $O(\log \Delta)$ min-plus products can essentially be replaced by only $O(\log \log n)$ such products. This does not change the value of the constant c.

THEOREM 4.2. *The all-pairs version of the Minimum-Cost Plan (MCP) problem in a graph that may contain negative cycles can be solved in $O(mn^2 + n^3 \log n)$ time. The same time bound applies when at most Δ rechargings can be used on each path.*

Proof. Dorfman et al. [4] gave algorithms for solving the all-pairs MFC and MIC problems in $O(mn^2 + n^3 \log n)$ time on graphs that may contain negative cycles. The MPP problem can be solved in $O(n^3)$ time using the naïve algorithm and the $APSP(n^2, n)$ problem can be solved in $O(n^3)$ time using the Floyd-Warshall algorithm [6, 12]. The $APSP_\Delta(n^2, n)$ can be solved in $O(n^3 \log \Delta) = O(n^3 \log n)$ time by solving $O(\log \Delta)$ MPP problems. $\quad \square$

All the time bounds above are in a model similar to the addition-comparison model used for standard shortest paths algorithms. (See, e.g., Zwick [15].) Arc and recharge costs can be arbitrary real numbers but it is assumed that 'reasonable' operations on them can be performed in constant time. Faster algorithms can be obtained if the battery capacity B is a relatively small integer, all arc costs $c(a)$ are integers, and all charging costs $r(v)$ are fairly small integers. In particular, if $R = \max\{r(v) \mid v \in V, r(v) < \infty\}$, then all arc lengths in the $APSP(n^2, n)$ instance that needs to be solved are in $\{0, 1, \ldots, BR\}$. Zwick [16] obtained an $O(M^{1/(4-\omega)} n^{2+1/(4-\omega)})$-time algorithm for the APSP problem when arc lengths are integers of absolute value at most M, where ω is the exponent of matrix multiplication. Currently $\omega < 2.3719$ due to a recent result of Duan, Wu and Zhou [5]. Note that the running time of this algorithm is subcubic when $M \ll n^{3-\omega}$, or in our case when $BR \ll n^{3-\omega}$. If $\omega > 2$, then an improved running time can be obtained using rectangular matrix multiplication. See Zwick [16] for the exact details.

Faster algorithms can also be obtained for the MFC and MIC problems in the word-RAM model, in which arc costs are assumed to be integers that fit into single machine words. This is done by replacing the standard comparison-based priority queues used by these algorithms by word-RAM priority queues. (See, e.g., Thorup [11].) The running times of the improved algorithms, when there are no negative cycles, are still $\Omega(mn)$. Thus, no improvement is obtained over the bound given in Theorem 4.1.

5 Reducing shortest paths to minimum-cost plans

In this short section we describe a trivial reduction from $APSP(m, n)$ to $MCP(m + n, 2n)$, showing that the Minimum-Cost Plans (MCP) problem is at least as hard as the classical All-Pairs Shortest Paths (APSP) problem.

THEOREM 5.1. *The $APSP(m, n)$ problem in a graph with nonnegative arc costs can be reduced in linear time to an $MCP(m + n, 2n)$ problem.*

Proof. Let $G = (V, A, c)$, where $c : A \to \mathbb{R}^+$ be the input to the $APSP$ problem. Construct a graph $G' = (V', A', c', r')$ where $V' = \{v', v \mid v \in V\}$ and $A' = \{(v', v) \mid v \in V\} \cup A$. Let $c'(v', v) = 0$, for every

---

[3] We simply need to raise the weighted adjacency matrix of the graph, with zeros on the diagonal, to the Δ-th power with respect to min-plus products. This can be easily done using $O(\log \Delta)$ min-plus products.

$v \in V$, and $c'(a) = c(a)$, if $a \in A$. Also, let $r(v') = 1$ and $r(v) = \infty$, for every $v \in V$. Let $B = n \max_{a \in A} c(a)$. We have $\delta^G(s,t) = \rho_B^{G'}(s',t)$, for every $s,t \in V$, where $\delta^G(s,t)$ is the distance from s to t in G and $\rho_B^{G'}(s',t)$ is the minimum cost of a plan from s' to t in G'. This follows since in G' it is only possible to charge the battery at the initial vertex and all arc costs are non-negative. Thus, a plan corresponds to a standard shortest path. $\square$

6 Non-zero initial charges

Up to this point, we considered optimal travel plans from s to t in which the car starts from s with an empty battery. (In particular, if no recharging is possible at s and all outgoing arcs of s have positive costs, the car cannot move out of s.) It is natural to also consider the case in which the car starts at s with some fixed initial charge a. (For example, $a = B$ corresponds to the case in which the car starts from s with a full battery.)

6.1 Fixed initial charges A simple reduction allows us to solve, within the same asymptotic time bounds, the all-pairs version of the *MCP* problem in which each vertex $v \in V$ has a fixed initial charge $0 \leq a(v) \leq B$ associated with it. The initial charges at different vertices are not necessarily the same.

For each original vertex $v \in V$ create a new vertex v_0 and add an arc (v_0, v) whose cost is $-a(v)$. Let $r(v_0) = \infty$, i.e., no recharging is possible at v_0. (Alternatively, we could let $r(v_0) = r(v)$.) Starting at v_0 with an empty battery corresponds to starting at v with a charge of $a(v)$. The new graph has $2n$ vertices and $m + n$ vertices. Thus, the asymptotic running times of the algorithms given in Section 4 are unaffected.

6.2 Adding an initial charge or a source vertex A further natural generalization is to allow an arbitrary initial charge a, not known in advance, at a specified source vertex s. We show that if there are no negative cycles in the graph, then after the all-pairs version of the *MCP* problem with zero initial charges is solved, we can solve the single-source version of the *MCP* problem from a given source vertex s and an initial charge a in $O(n^2)$ time.

The problem above corresponds to adding a new vertex s_0 with $r(s_0) = \infty$, i.e., no charging at s_0, and an arc (s_0, s) of cost $-a$ and solving the single-source version of the *MCP* problem from s_0 with an initial charge of 0. We actually solve the more general problem in which we add new vertex s_0 and a collection of arcs $(s_0, v_1), (s_0, v_2), \ldots, (s_0, v_k)$ to an arbitrary number of original vertices of the graph, each with possibly different cost $c(s_0, v_i) = -a_i \leq 0$. This corresponds to the option of using one of several available electric cars, where the i-th available car is at vertex v_i and has an initial charge of a_i.

We present a reduction from the problem of adding a new source vertex to a graph on which the all-pair *MCP* problem was already solved, to the solution of two single-source *MFC* problems and two vector-matrix min-plus products. When there are no negative cycles in the graph, the two single-source *MFC* problems can be solved in $O(m + n \log n)$ time, using a feasible potential function computed while solving the all-pairs *MFC* and *MIC* problems on the original graph. (See [4] for details.) The two $1 \times 2n$ by $2n \times 2n$ min-plus products can be computed naïvely in $O(n^2)$ time.

We start by computing $\alpha_{B,0}(s_0, v), \alpha_{B,B}(s_0, v)$ for every $v \in V$. These are the two instances of the single-source *MFC* problem.

Next, as in Section 3.3, we compute $\ell_B(s_0^0, v^b)$, for every $v \in V$ and $b \in \{0, B\}$. Recall that $\ell_B(s_0^0, v^b)$ is the minimum cost of getting from s_0 to v, starting with an empty battery and arriving with either an empty or full battery and using at most one recharging. This can be done in $O(n^2)$ time by naïvely computing a vector-matrix min-plus product.

Now, since we already know $\delta_{\ell_B}(u^a, v^0)$, for every $u, v \in V$ and $a \in \{0, B\}$, we can compute $\rho_B(s, v)$, for every $v \in V$ using an additional vector-matrix min-plus product, again in $O(n^2)$ time. More concretely, for every $v \in V$, we have $\rho_B(s_0, v) = \min_{u \in V, a \in \{0, B\}} \ell_B(s_0^0, u^a) + \delta_{\ell_B}(u^a, v^0)$. Correctness follows again by Lemma 3.2.

7 Concluding remarks

We have presented a simple reduction from the problem of computing Minimum-Cost Plans (*MCP*) between all pairs of vertices in a graph to two simpler problems: The first is the computation of optimal energetic paths when no recharging of the battery is allowed. This problem comes in two equivalent variants: Maximum Final Charge (*MFC*) and Minimum Initial Charge (*MIC*). The second problem is the standard All-Pairs Shortest Paths problem.

Using this simple reduction we have obtained an $O(\frac{n^3}{2^{c\sqrt{\log n}}} + mn)$-time algorithm for solving the *MCP* in

graphs with no negative cycles. This matches the running time of the fastest APSP algorithm of Williams [13, 14], unless m is extremely close to n^2.

An interesting open problem is whether there is an $\tilde{O}(mn)$-time algorithm for the all-pairs MCP problem in n-vertex, m-arc graphs, essentially matching the complexity of the standard APSP problem for sparse graphs.

Another interesting open problem is whether there is an $\tilde{O}(n^3)$-time algorithm for the all-pairs MCP problem when the input graph may contain negative cycles. (An interesting feature of the MCP problem is that minimum-cost plans are well-defined, and are of finite length, even in this case. They may not be simple, however.)

All algorithms presented assume that B, the maximum capacity of the battery, and Δ, the maximum number of rechargings allowed on each path, are fixed and known in advance. Is it possible to obtain an efficient algorithm that preprocesses an input graph and is then able to quickly answer queries of the form $\rho_{B,\Delta}(s,t)$, i.e., what is the minimum cost of a plan from s to t when the capacity of the battery is B and at most Δ rechargings are allowed on the way from s to t?

Finally, we remark that we considered the natural problem of finding minimum-cost plans, ignoring the time it takes to traverse arcs or to charge the battery. Finding a plan that can be implemented within a given time limit is easily seen to be a NP-hard problem by a reduction from 0-1 knapsack. Similarly, the problem becomes NP-hard if each arc (u,v) has a monetary cost $r(u,v)$, i.e., a *toll* that should be paid to traverse the arc, in addition to its energetic cost $c(u,v)$, and the goal is to minimize the cost of travelling from s to t. It may be possible to obtain interesting approximation algorithms, but this is beyond the scope of the current paper.

Acknowledgement

We would like to thank two anonymous SOSA reviewers for valuable comments that helped improve the paper.

References

[1] Alfred V. Aho, John E. Hopcroft, and Jeffrey D. Ullman. *The design and analysis of computer algorithms*. Addison-Wesley, 1974.

[2] Timothy M. Chan and R. Ryan Williams. Deterministic APSP, orthogonal vectors, and more: Quickly derandomizing Razborov-Smolensky. *ACM Trans. Algorithms*, 17(1):2:1–2:14, 2021.

[3] Edsger W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[4] Dani Dorfman, Haim Kaplan, Robert E. Tarjan, and Uri Zwick. Optimal energetic paths for electric cars. In *Proc. of 31st ESA*, 2023. See also CoRR, abs/2305.19015.

[5] Ran Duan, Hongxun Wu, and Renfei Zhou. Faster matrix multiplication via asymmetric hashing. *CoRR*, abs/2210.10173, 2022. To appear in FOCS 2023.

[6] Robert W. Floyd. Algorithm 97: Shortest path. *Communications of the ACM*, 5(6):345, 1962.

[7] Michael L. Fredman and Robert Endre Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the ACM*, 34(3):596–615, 1987.

[8] Thomas Dueholm Hansen, Haim Kaplan, Robert E. Tarjan, and Uri Zwick. Hollow heaps. *ACM Trans. Algorithms*, 13(3):42:1–42:27, 2017.

[9] Samir Khuller, Azarakhsh Malekian, and Julián Mestre. To fill or not to fill: The gas station problem. *ACM Transactions on Algorithms (TALG)*, 7(3):1–16, 2011.

[10] Seth Pettie. A new approach to all-pairs shortest paths on real-weighted graphs. *Theor. Comput. Sci.*, 312(1):47–74, 2004.

[11] Mikkel Thorup. Integer priority queues with decrease key in constant time and the single source shortest paths problem. *J. Comput. Syst. Sci.*, 69(3):330–353, 2004.

[12] Stephen Warshall. A theorem on boolean matrices. *Journal of the ACM*, 9(1):11–12, 1962.

[13] R. Ryan Williams. Faster all-pairs shortest paths via circuit complexity. *SIAM J. Comput.*, 47(5):1965–1985, 2018.

[14] R. Ryan Williams. From circuit complexity to faster all-pairs shortest paths. *SIAM Rev.*, 63(3):559–582, 2021.

[15] Uri Zwick. Exact and approximate distances in graphs - A survey. In *Proc. of 9th ESA*, volume 2161 of *Lecture Notes in Computer Science*, pages 33–48. Springer, 2001.

[16] Uri Zwick. All pairs shortest paths using bridging sets and rectangular matrix multiplication. *J. ACM*, 49(3):289–317, 2002.