Proceedings of the Federated Conference on
Computer Science and Information Systems pp. 349–356

ISBN 978-83-60810-51-4

Online Algorithm for Battery Utilization in Electric Vehicles

Ron Adany
Computer Science Department
Bar-Ilan University
Ramat-Gan 52900, Israel
Email: adanyr@cs.biu.ac.il

Tami Tamir
School of Computer Science
The Interdisciplinary Center
Herzliya, Israel
Email: tami@idc.ac.il

***Abstract*—We consider the problem of utilizing the pack of batteries serving current demands in Electric Vehicles. When serving a demand, the current allocation might be split among the batteries in the pack. Due to its internal chemistry structure, a battery's life depends on the discharge current used for supplying the requests. Any deviation from the (a-priori known) optimal discharge-current is associated with a penalty. Thus, the problem is to serve an online sequence of requests for energy, in a way that minimizes the total penalty associated with the service.**

We first formulate the problem as a combinatorial optimization problem. We show that the problem is strongly NP-hard and is hard to approximate within an additive gap of $\Omega(m)$ from the optimum, where m is the number of batteries in the pack. This hardness result is valid even in the offline case, where the sequence of current demands is known in advance. For the online problem, we suggest an algorithm in which the total penalty might be larger than the minimal possible by at most an additive gap of $1.5m$ - independent of the initial capacity of the batteries and the number of requests in the sequence. Finally, we provide a lower bound of 1.5 for the multiplicative competitive ratio of any online algorithm.

To the best of our knowledge, our work is the first to analyze the problem theoretically.

I. Introduction

IN THE last few years the idea of electrically powered cars has turned into reality. This old idea, from the early years of the automobile industry in the late 1890's, is now a real alternative to the gasoline powered vehicles [1], [2]. Electric Vehicles (EVs) are currently considered the next generation of cars in the world of automobiles.

One of the most critical components of EV is the rechargeable battery [3]. The use of a battery as an energy source raises several issues such as limited driving ranges and high costs. Batteries are very expensive [4] and thus it is critically important to extend their life as much as possible. The battery's life is affected by the way the energy is discharged (allocated), which is the focus of this paper. It is important to distinguish between the battery's *capacity* and the battery's *life*. The battery's *capacity* is the amount of energy that can be stored in the battery (and later discharged), i.e., the amount of energy of one charge-discharge cycle. The battery's *life* is the total amount of energy that can be extracted from all charge-discharge cycles of the battery [5]. In other words, the battery's *life* is the overall energy output of the battery before it fails to meet specific performance criteria.

The EV battery is actually a pack of cells that are connected in serial and in parallel in order to provide the voltage and current required for propulsion. Cells are serially connected in order to provide the required voltage, and the series are connected in parallel to provide the required current. Our work considers the energy allocation provided by the individual cell-series in the batteries. For simplicity, we denote each cell-series as a battery.

The most dominant factor of an EV battery's life is the discharge current used. Each battery's chemistry is designed to be discharged in a specific current, while higher or lower currents have negative effects on it [6], [7], [8]. Moreover, as demonstrated in [9], an optimal discharge current exists for each battery and depends on the specific chemistry. Based on these insights, we propose a penalty function that maps each discharge current to a numeric value reflecting its detrimental affect on the battery's life.

The common discharge method in EV is very simple and naive, where each current demand is supplied using all the batteries in the pack, and the load is equally divided. The rational behind this method is simplicity of implementation, keep all batteries in the same condition (balanced) and the assumption that the lower the discharge current the better. However, as described above, the behavior and performance of a real battery is more complex.

A. Our Results

Motivated by the possibility of extending the EV batteries' life by a smart operation, we proposed an advanced online energy allocation algorithm. We also analyze the problem theoretically and provide results regarding its complexity status under common theoretical measures.

The performance of an energy allocation algorithm is evaluated according to its gap from an optimal allocation, i.e., the gap from an allocation that serves all current demands with the minimal possible penalty. We provide hardness results for both the offline problem, where the current demands are known in advance, and for the online problem, where the sequence of current demands is unknown in advance. In practice, demands should be served without apriori knowledge of forthcoming demands. Therefore, the main algorithm we suggest is for the online problem. Clearly, all hardness results we provide for

the offline problem also capture the online one. For the online problem we also provide a lower bound on the competitive ratio of any deterministic algorithm.

A formal, mathematical, description of the problem is given in Section II. In Section III we consider the offline problem. We show that the problem of serving the requests with the minimal possible penalty is strongly NP-hard and hard to approximate within an additive gap of $\Omega(m)$ from the optimum.

We then consider the online problem. In Section IV we suggest an algorithm in which the total penalty might be larger than the minimal possible one by at most an additive gap of $1.5m$ - independent of the number of requests in the sequence and of the initial capacity of the batteries. Our result is optimal, i.e., associated with the minimal possible penalty, for a significant prefix of the sequence – as long as the remaining capacity of the batteries is not below some threshold. Thus, our algorithm is optimal in the sense that it guarantees minimal damage to the batteries' life when drivers charge their EV frequently enough and do not wait until the batteries are empty. Finally, in Section V we provide a lower bound of 1.5 for the competitive ratio of any online algorithm. Recall, an online algorithm has a competitive-ratio c if, for every possible instance, its objective value is within a factor of c from the optimum.

We note that our results can be applied to other resource allocation problems in which there are m servers (machines, batteries) that should serve n clients (jobs, current demands). Each client is associated with a request for some amount of resource. A request can be satisfied by several servers, i.e., the service of a request might split. There is a penalty associated with each allocation and the objective is to minimize the total penalty. This general problem appears in many domains. For example, in Human Resources (HR), it is reasonable to allocate tasks to workers such that each worker is assigned a specific workload, and any deviation from this workload causes some penalty, specifically, high workloads conflict with working time regulations, and low workloads might be unprofitable.

Due to space constraints, some of the proofs are omitted from this extended abstract.

B. Related Work

Battery management and power allocation has been widely studied. Most of the previous work has been aimed at maximizing the battery's lifetime, i.e., the time until the battery is empty and needs to be recharged.

Algorithms for extending batteries' lifetime by various discharge allocation schemes were discussed in [10] and [11]. Several discharge schemes were proposed including: (i) serial – discharging of a single battery each time until it is emptied and discharging next battery, (ii) static switching – discharging of a single battery for a certain amount of time, (iii) dynamic switching – discharging of each battery for a different amount of time depending on its physical state, e.g., remaining capacity, (iv) parallel – discharging of all batteries together where the workload splits equally. In simulations and lab experiments the static and dynamic algorithms increased the batteries's lifetime significantly. In addition, the lifetime of batteries operated by both static and dynamic switching algorithms increased as the switching frequency increased.

The energy demands in EVs are not known in advance and may be estimated using prediction methods based on driving profiles, driving stories and the history of demands. In [12] discharge methods among multiple batteries were discussed where the current requests and their distribution over time are known. We do not deal with this type of problem since we propose an online algorithm.

II. Problem Definition

The system consists of m identical batteries, $B_1, B_2, \ldots, B_m$, all having the same initial capacity C. There are n current demands, i.e. requests for current, given as an online sequence $d_1, d_2, \ldots, d_n$, where d_i is the energy required to satisfy the i-th request. The sequence, and in particular its length n, is unknown in advance, but it is guaranteed that the total capacity of the batteries can satisfy all requests, that is, $\sum_i d_i \leq mC$.

The i-th request can be satisfied in various ways. Any allocation of d_i from the batteries is acceptable, and there are no constraints regarding the distribution of d_i among the batteries. However, there are 'good' and 'bad' allocations – since an optimal discharge current exists.

Based on our understanding of the electrochemical properties of the individual cells [6], [7], [8], we define a penalty function that reflects the damage to the battery's life from the discharge current. The actual penalty function depends on the specific chemistry of the battery. However, it is reasonable to assume the following basic structure. Let I_{OPT} be the optimal discharge current, whose supply results in maximization of the battery life. The penalty function should fulfill the following conditions.

1) The penalty for no discharge is 0.
2) The penalty for supplying I_{OPT} is 0.
3) All other discharge currents have positive penalty values, according to their distance from I_{OPT}.

We assume the following linear penalty function, described in Fig. 1. Let x be the discharge current, then, for some parameter $\alpha > 0$,

$$Penalty(x) = \begin{cases} 0, & \text{if } x = 0 \text{ or } x = I_{OPT} \\ \alpha |I_{OPT} - x| & \text{else} \end{cases} \quad (1)$$

By simple scaling of all demands and capacities, we assume throughout this paper, w.l.o.g., that $I_{OPT} = 1$ and $\alpha = 1$. We denote by C_j^i the remaining capacity of battery B_j before current demand i. The goal is to determine the values x_j^i, where x_j^i is the discharge current (power allocation) of battery B_j to supply the current demand i.

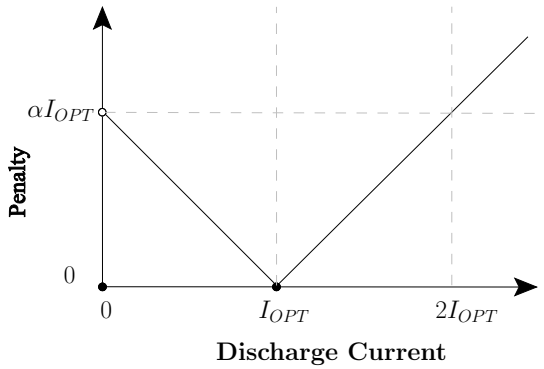

Fig. 1. The penalty function.

III. The Offline Problem

In this section we consider the case in which all requests are known in advance. While this is not the case in practice, from the theoretical aspect it is interesting as any hardness result for this model also captures the online problem. The problem can be described as follows.

$$\begin{aligned}
\min \quad & \sum_{i=1}^{n}\sum_{j=1}^{m} Penalty(x_j^i) & \\
\text{s.t.} \quad & \sum_{i=1}^{n} x_j^i \le C & \forall j = 1 \dots m \\
& \sum_{j=1}^{m} x_j^i = d_i & \forall i = 1 \dots n \\
& 0 \le x_j^i, \quad x_j^i \in R & \forall i = 1 \dots n, \quad j = 1 \dots m
\end{aligned}$$

A. Hardness of the Offline Problem

We first prove that the offline problem of serving all requests with the minimal possible penalty is strongly NP-hard. Next, we show that it is NP-hard to approximate the minimal possible penalty within an additive $\Omega(m)$ factor.

Theorem III.1. *The problem of serving all requests with the minimal possible penalty is strongly NP-hard even if all current demands are known in advance.*

Proof: We show a reduction from the 3-*partition* problem. The input to 3-*partition* is a set of $n = 3m$ numbers $S = \{d_1, \dots, d_{3m}\}$ such that $\forall i,\ \frac{1}{4} \le d_i \le \frac{1}{2}$ and $\sum_{i=1}^{n} d_i = m$. The goal is to divide S into m subsets $S_1, \cdots, S_m$ of S such that $\sum_{d_k \in S_j} d_k = 1$ for all $1 \le j \le m$. Note that each such set consists of exactly three numbers. The 3-*partition* problem is known to be strongly NP-hard [13].

Given S, we construct the following instance of our problem: m batteries, each with an initial capacity of $C = 1$, and $n = 3m$ current demands, where the i-th request is for d_i.

Claim III.2. *The set S has a 3-partition if and only if the minimum penalty for serving the requests is $2m$.*

Proof: Given a 3-*partition* of S, let $S_1, \cdots, S_m$ be the required partition, i.e., $\sum_{d_k \in S_j} d_k = 1$ for all $1 \le j \le m$. We serve the requests as follows: for every subset S_j, assume that $S_j = \{d_{j_1}, d_{j_2}, d_{j_3}\}$, then the corresponding three requests are served from battery j. All requests are less than 1, i.e., $d_i \le 1$, thus the penalty from each is $1 - d_i$. Accordingly, the total penalty for serving requests from battery j is $(1-d_{j_1}) + (1-d_{j_2}) + (1-d_{j_3})$, and the total penalty for serving all $3m$ requests, as induced by the m sets is $\sum_{j=1}^{m}[(1-d_{j_1}) + (1-d_{j_2}) + (1-d_{j_3})] = 3m - \sum_{i=1}^{n} d_i = 3m - m = 2m$.

For the other direction, assume that the requests are served such that the total penalty is $2m$. Since all requests for current demand are less than 1, the penalty for serving request i is exactly $m_i - d_i$, where $m_i \ge 1$ is the number of batteries for which $x_j^i > 0$ (that allocate some energy to request i). Therefore, the total penalty is $\sum_{i=1}^{n}(m_i - d_i) = \sum_{i=1}^{n} m_i - m$. In order for this sum to be $2m$, it must hold that $\sum_{i=1}^{n} m_i = 3m$. Since $n = 3m$ and m_i values are non-negative integer numbers, $m_i = 1$ must be true for all i. In other words, each request is supplied only by a single battery. Also, the range of d_i, i.e., $\frac{1}{4} \le d_i \le \frac{1}{2}$, implies that exactly three demands are served by each battery. Since the total current demand is exactly the total available energy, i.e., $\sum_{i=1}^{n} d_i = \sum_{j=1}^{m} C$, it must be that for every $1 \le j \le m$, the requests that are served by the j-th battery constitute a total demand of exactly $C = 1$ and the mapping of requests to the batteries that serve them induces a 3-partition. ■

■

Our next hardness proof shows that the gap between the total penalty of any efficient algorithm, and the penalty of an optimal algorithm is $\Omega(m)$. We use an idea suggested in a hardness proof for the problem of packing with item fragmentation [14].

Theorem III.3. *For any $m > 1$, a set of requests exists such that for some $P \ge 0$, the optimal serve of all the requests by m batteries constitute a total penalty P, while service of all the requests by m batteries with a total penalty less than $P + \lfloor \frac{m}{2} \rfloor$ is NP-hard.*

Proof: Recall that $m_i \ge 1$ denotes the number of batteries participating in the service of request i, i.e., $m_i = |\{j | x_i^j > 0\}|$.

The proof is based on defining a set of requests, i.e., an instance σ, with the following attributes:

- (A_1) All the requests are small, i.e., for all i, $d_i < 1$.
- (A_2) In the optimal solution, every request for current demand d_i is provided by a single battery, i.e., for all i, $m_i = 1$.
- (A_3) If $P \ne NP$ then in the allocation determined by any efficient algorithm, $\sum_i m_i \ge n + \lfloor m/2 \rfloor$.

Note that the penalty for any allocation of $x_i^j < 1$ is $1 - x_i^j$. Thus, the total penalty for serving request i is $m_i - d_i$. Given that σ fulfills attributes (A_1) and (A_2), we conclude that the minimal penalty achieved by an optimal allocation is $P_{OPT}(\sigma) = \sum_{i=1}^{n}(m_i - d_i) = n - \sum_i d_i$. Also, by attribute (A_3), the total penalty of any efficient algorithm is at least $P_{ALG}(\sigma) = \sum_i (m_i - d_i) \ge n + \lfloor m/2 \rfloor - \sum_i d_i$. Therefore, $P_{ALG}(\sigma) - P_{OPT}(\sigma) \ge \lfloor m/2 \rfloor$.

We now describe the construction of an instance σ that fulfills $(A_1)-(A_3)$. For this, we assume that the batteries have different initial capacities and that m is even. Later we explain how these assumptions can be removed.

In order to achieve attribute (A_3), we use a reduction from the *Partition* problem. The input for *Partition* is a set $U = \{a_1,\ldots,a_h\}$ of positive integers with a total size of $2S$. The goal is to determine whether a subset $U' \subseteq U$ of a total size S exists.

For a given number of batteries, m, and a given instance of *Partition*, $U=\{a_1,\ldots,a_h\}$, construct an instance for the energy allocation problem with m batteries and $n = h\cdot m/2$ current demands. The current demands consists of $k = m/2$ sets, $R_0,\ldots,R_{k-1}$ each comprising h requests. Let $M > (2S+1)$ be an integer. The set R_0 comprises requests for current demands $a_1, a_2, \ldots, a_h$; R_1 comprises requests for current demands $a_1M, a_2M, \ldots, a_hM$, and in general, R_ℓ comprises requests for current demands $a_1M^\ell, a_2M^\ell, \ldots, a_hM^\ell$. The battery capacities are $S, SM, SM^2, \ldots, SM^\ell, \ldots, SM^{k-1}$ and there are two batteries of each capacity. Note that the total required energy is exactly the total capacity of the batteries. Finally, let $I_{OPT} = SM^k$. For simplicity of the description, the attributes are described assuming $I_{OPT} = 1$, this is w.l.o.g. as it is possible to scale all demands and capacities by a factor of $1/SM^k$.

If a partition of the items in U into two sets of size S exists, then an energy allocation fulfilling (A_2) exists. Such an optimal allocation serves all the requests of R_ℓ by the two batteries with a capacity of SM^ℓ. Since the total demand of the requests in R_ℓ is exactly $2SM^\ell$ and a partition exists, R_ℓ can be partitioned into two sets of requests, each set with total demand of SM^ℓ, and it is possible to serve all the requests such that every request is served by a single battery.

For the other direction, consider any service of the demands. Since the total required energy is exactly the total capacity of the batteries, for every battery j with SM^ℓ capacity, $\sum_i x_i^j = SM^\ell$ holds.

Claim III.4. *If some battery, B_j, serves only full requests, that is, for all i either x_i^j is 0 or d_i, then a partition of U exists.* ■

We conclude that if $P \neq NP$ then, in any allocation provided by an efficient algorithm, every battery will serve at least one partial request. Thus, there are at least $m/2$ fractions of requests and attribute (A_3) holds.

In order to extend the proof to instances with identical batteries, we set the capacities of all $2k$ batteries to SM^k and add two *filler requests* of size $S(M^k - M^\ell)$ to each set of demands R_ℓ. Nonetheless, the total required energy is still exactly the total capacity of the batteries. Note that the two smallest filler requests constitute a total size of $2S(M^k - M^{k-1})$, which is larger than SM^k for any $M > 2$. Therefore, in any allocation of full requests, each battery serves at most one filler request. Furthermore, the total demand of non-filler request is too small to fully utilize any battery, therefore any fully utilized battery must serve exactly one filler request. Assume that some battery serve only full requests and let $S(M^k - M^z)$ be the demand of the filler request served by the battery. The rest of the energy is allocated to requests of a total demand SM^z and the proof for the different capacity batteries can be applied.

If m is odd, we can add a single request for a current demand of SM^k, and setting $k = \lfloor m/2 \rfloor$, we attain an instance for which the above reduction can be applied. ■

IV. An Almost Optimal Online Algorithm

In this section we describe an online power allocation scheme that is guaranteed to serve all requests in an instance σ with a penalty of at most $P_{OPT}(\sigma)+1.5m$. By theorem III.3, a lower bound of $P_{OPT}(\sigma)+\Omega(m)$ also applies to the offline case. Therefore, our algorithm is optimal in the sense that only the constant factor in the additive term might be reduced. Another property of our algorithm is that for a significant prefix of the requests, an optimal allocation is guaranteed. This property implies that our algorithm guarantees minimal damage to the batteries' life of drivers who charge their EV frequently enough and do not wait until the batteries are empty. The last demands, i.e., the demands when the batteries are almost empty, turned out to be the most challenging, as the amount of energy left in each battery may be small and an optimal discharge current might not be possible.

The proposed solution distinguishes between two possible statuses of the batteries. As long as the status is $AboveTheLine$ (to be defined later), the serve of all requests is associated with the minimal possible penalty. Once the status is not $AboveTheLine$, our allocation might involve a redundant penalty. However, it is possible to bound this redundant penalty by $1.5m$.

In order to determine the batteries' status before serving request i, we sort the batteries in non-increasing order of remaining capacity, i.e., $C_1^i \geq C_2^i \ldots \geq C_m^i$.

Definition IV.1. *If $C_1^i \geq 1.5$ and $C_j^i \geq 1$ for every $j > 1$, then the batteries status before request i is $AboveTheLine$.*

Algorithm 1 describes the energy allocation scheme for a single request. For simplicity, as our analysis refers to a single request, d_i, we drop the index i whenever it is clear from the context. Specifically, d is the request, C_j is the remaining capacity of B_j before the request, and x_j is the power allocated to the request by B_j.

In some cases the algorithm allocates energy *in a balancing way* among a selected set of batteries. That is, the energy is allocated from the batteries in the set with the highest remaining capacity, such that $x_1 \geq x_2 \geq \ldots$, while trying to balance the remaining capacities in the set after the allocation. This can be done by first allocating from B_1, till $C_1 = C_2$, then allocating evenly from both B_1, B_2, and so on, until the whole demand is allocated.

In some cases the algorithm calls the procedure *Greedy-FinalAllocation*, given in Algorithm 2. In this procedure the allocation is done greedily from the battery with the lowest

remaining capacity, moving to the next lowest battery when the battery is emptied.

We state that a request for energy d has a *low fraction* if $d - \lfloor d \rfloor < \frac{1}{2}$. Otherwise, the request has a *high fraction*. Note that a request has a low fraction if and only if $d - \lfloor d \rfloor < \lceil d \rceil - d$. For example $d = 2.3$ has a low fraction while $d = 2.8$ has a high fraction.

The analysis of the algorithm is based on several observations. First, we show that as long as the batteries' status is $AboveTheLine$, then the service of all requests is associated with the minimal possible penalty. Next, we bound the total possible penalty once the batteries' status is not $AboveTheLine$.

Lemma IV.2. *As long as the batteries' status is $AboveTheLine$, the service of all requests is associated with the minimal possible penalty.*

Proof: Consider a current demand, d, and use $P_{OPT}(d)$ to denote the minimal possible penalty for supplying it.

If $d \leq 1$ then $P_{OPT}(d) = 1 - d$, achieved by allocating all energy from a single battery - as in our algorithm (line 4). If $d \geq m$, then $P_{OPT}(d) = d - m$, achieved by allocating at least 1 from each of the m batteries. Such an allocation is done in our algorithm (lines 10-11). Given that $C_j^i \geq 1$ for every $j > 1$, the two above allocations are possible.

If $1 < d < m$, then the minimal possible penalty is the distance of d from the nearest integer, given by $P_{OPT}(d) = \min\{d - \lfloor d \rfloor, \lceil d \rceil - d\}$. In both cases, this value is at most $\frac{1}{2}$. If the request has a low fraction then penalty $P_{OPT}(d)$ can be achieved by allocating at least 1 from $\lfloor d \rfloor$ batteries. If the request has a high fraction then penalty $P_{OPT}(d)$ can be achieved by allocating at least $\frac{1}{2}$ and at most 1 from $\lceil d \rceil$ batteries. Such an allocation is done in our algorithm (lines 19-20 and 27-28 respectively). Note that since the status is $AboveTheLine$, i.e., $C_1^i \geq 1.5$ and $C_j^i \geq 1$ for every $j > 1$, the above allocations are possible. Therefore, if the batteries' status before the allocation is $AboveTheLine$, then it is always possible to follow the suggested allocation which is associated with the minimal possible penalty. ■

We conclude that the algorithm might cause a redundant penalty only in *GreedyFinalAllocation*. In order to bound the total redundant penalty, we first bound the gap between the remaining capacities of any pair of batteries.

Claim IV.3. *For any request i and two batteries, j, j', $|C_j^i - C_{j'}^i| < 1.5$.*

Proof: The proof is by induction on the number of requests serviced. Initially, before the first request, all capacities equal C and the claim is clearly valid. Assume that the claim holds before the allocation of the i-th request, we show it is valid also afterward. Let S denote the set of batteries allocating power to the i-th request, i.e., $k \in S$ if and only if $x_k^i > 0$.

Consider first the case that the batteries' status is $AboveTheLine$. If both $j, j' \notin S$, then clearly, $C_j^i = C_j^{i+1}$ and $C_{j'}^i = C_{j'}^{i+1}$, so the gap between the remaining capacities remains the same. If both $j, j' \in S$, since the allocation is

Algorithm 1 Energy Allocation for a single request

Input: Power demand, d, battery capacities, $C_1 \geq \ldots \geq C_m$.

1: Let $\hat{m}$ be the number of batteries with a capacity of at least 1, i.e., $\hat{m} = \max_j C_j \geq 1$.
2: **if** $d \leq 1$ **then**
3: **if** $d \leq C_1$ **then**
4: Let $x_1 = d$.
5: **else**
6: CALL: *GreedyFinalAllocation*.
7: **end if**
8: **else if** $d \geq m$ **then**
9: **if** batteries status is $AboveTheLine$ **then**
10: Determine initial allocation of $x_j = 1$ from batteries $j = 1, \ldots, m$.
11: Allocate the remaining current demand, in a balancing way among batteries $j = 1, \ldots, m$.
12: **else** {batteries status is not $AboveTheLine$}
13: Determine initial allocation of $x_j = 1$ from batteries $j = 1, \ldots, \hat{m}$.
14: Let $x_j = C_j$ for all $j = \hat{m} + 1, \ldots, m$.
15: Allocate the remaining current demand, i.e., $d - \hat{m} - \sum_{j=\hat{m}+1}^{m} C_j$, by *GreedyFinalAllocation*.
16: **end if**
17: **else if** $d - \lfloor d \rfloor < \frac{1}{2}$ **then** {low fraction request}
18: **if** $\lfloor d \rfloor \leq \hat{m}$ and $d \leq \sum_{j=1}^{\lfloor d \rfloor} C_j$ **then**
19: Determine initial allocation of $x_j = 1$ from batteries $j = 1, \ldots, \lfloor d \rfloor$.
20: Allocate the remaining current demand, i.e., $d - \lfloor d \rfloor$, in a balancing way from batteries $j = 1, \ldots, \lfloor d \rfloor$.
21: **else**
22: Determine initial allocation of $x_j = 1$ from batteries $j = 1, \ldots, \min\{\hat{m}, \lfloor d \rfloor - 1\}$.
23: Allocate the remaining current demand, i.e., $d - \min\{\hat{m}, \lfloor d \rfloor - 1\}$, by *GreedyFinalAllocation*.
24: **end if**
25: **else** {high fraction request}
26: **if** $\lceil d \rceil \leq \hat{m}$ **then**
27: Determine initial allocation of $x_j = \frac{1}{2}$ from batteries $j = 1, \ldots, \lceil d \rceil$.
28: Allocate the remaining current demand, i.e., $d - \lceil d \rceil / 2$, in a balancing way from batteries $j = 1, \ldots, \lceil d \rceil$ limiting the allocation from any single battery by 1, i.e., $x_j \leq 1$ for all j.
29: **else**
30: Determine initial allocation of $x_j = 1$ from batteries $j = 1, \ldots, \min\{\hat{m}, \lfloor d \rfloor\}$.
31: Allocate the remaining current demand, i.e., $d - \min\{\hat{m}, \lceil d \rceil - 1\}$, by *GreedyFinalAllocation*.
32: **end if**
33: **end if**

Algorithm 2 GreedyFinalAllocation

Input: Power demand, d, battery capacities, $C_1 \geq \ldots \geq C_m$.

1: $index = m$
2: **while** $d > 0$ **do**
3: Set $x_{index} = \min\{C_{index}, d\}$
4: $d = d - x_{index}$
5: $index = index - 1$
6: **end while**

done in a balancing way, i.e., $x^i_j \geq x^i_{j'}$, the gap between the remaining capacities can only decrease. The third case, i.e., $j \in S$ and $j' \notin S$, can only happen if $d_i < m$. According to the algorithm the maximal allocation of a single battery in this case is less than 1.5.

When the batteries' status is not $AboveTheLine$, either the maximal capacity of a battery is less than 1.5, which clearly implies that the maximal gap is less than 1.5, or the allocation is done in a balancing way as explained above. ■

Next, we bound the total redundant penalty after the batteries status is not $AboveTheLines$.

Theorem IV.4. *The total penalty of the algorithm might be larger than the minimal possible penalty by at most* $1.5m$.

Proof: Use $P_{OPT}(d_i), P_{ALG}(d_i)$ to denote the minimal possible penalty, and the penalty caused by the algorithm for the i-th request. Let $\Delta(i) = P_{ALG}(d_i) - P_{OPT}(d_i)$ be the redundant penalty which occurred in the service of request i; e_i be the number of batteries emptied as a result of servicing request i; and m' denote the number of batteries with a positive remaining capacity after all requests are served, that is, $m = m' + \sum_i e_i$. We show that $\sum_i \Delta(i) \leq 1.5m$ by showing that $\sum_i \Delta(i) \leq \sum_i 1.5(e_i + m')$. By Lemma IV.2, as long as the batteries status is $AboveTheLine$, $\Delta(i) = 0$ holds. Therefore, we need to bound the redundant penalty after the batteries status is not $AboveTheLine$. Our analysis is based on combining the following three claims.

1) For all (except maybe for one) requests in which $e_i > 0$ it holds that $\Delta(i) \leq 1.5e_i$.
2) For a single request for which $e_i > 0$ it might be that $1.5e_i < \Delta(i) \leq 1.5(e_i + 1)$.
3) Either $m' > 0$, or for the last request, $e_i > 0$ and $\Delta(i) \leq 1.5(e_i - 1)$.

Before proving the above claims we note several observations. First, if a battery is emptied while the batteries' status is $AboveTheLines$ then by Lemma IV.2, $\Delta(i) = 0$. We therefore need to analyze only the cases in which the batteries are emptied when the batteries' status is not $AboveTheLines$. In this case, the algorithm might have redundant penalty only when *GreedyFinalAllocation* is called.

The batteries status is not $AboveTheLine$ if one of the two following condition is valid

1) $C^i_1 < 1.5$.
2) $C^i_m < 1$.

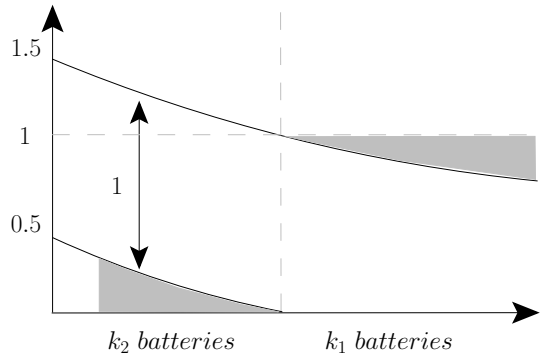

Fig. 2. The batteries status is not $AboveTheLine$ and $C^i_1 < 1.5$.

Claim IV.5. *If* $C^i_1 < 1.5$ *and* GreedyFinalAllocation *is called, then* $\Delta(i) \leq 1.5e_i$.

Proof: Assume that *GreedyFinalAllocation* empties k_1 batteries for which $C^i_j \leq 1$, and k_2 batteries for which $C^i_j > 1$ (as shown in Fig. 2). The penalty for the k_1 batteries is $\sum_j (1 - C^i_j) < k_1$. The penalty for the k_2 batteries is $\sum_j (C^i_j - 1) < 0.5k_2$. The last inequality follows from the fact that $C^i_j \leq C^i_1 < 1.5$. The total penalty is therefore $P_{ALG}(d_i) < k_1 + 0.5k_2$. All batteries participating in *GreedyFinalAllocation* except perhaps the last one are emptied, thus, $e_i = k_1 + k_2 - 1$.

We need to show that $k_1 + 0.5k_2 \leq 1.5(k_1 + k_2 - 1)$. This is equivalent to showing $k_1 + 2k_2 \geq 3$. Clearly, if $(k_1, k_2) \geq (1, 1)$ it is valid. We analyze the other cases separately.

Assume $k_2 = 0$, that is, the whole allocation is from batteries having a remaining capacity of at most 1. We show that for this case $\Delta(i) \leq e_i$. We distinguish between several cases, according to the value of d_i. If $d_i \leq 1$ then $P_{OPT}(d_i) = 1 - d_i$. Recall that the algorithm allocates d_i greedily, from k_1 batteries, out of which at least $k_1 - 1$ are emptied. Thus, $P_{ALG}(d_i) = \sum_{\{j|x_j>0\}} (1 - x_j) = k_1 - d_i$. We find that $\Delta(i) = k_1 - d_i - (1 - d_i) = k_1 - 1$, i.e., $\Delta(i) \leq e_i$.

If $1 < d_i < m$ and d_i has a low fraction. Let $w = d_i - \lfloor d_i \rfloor$. It holds that $P_{OPT}(d_i) = w$. The algorithm allocates $g_i = d_i - \min\{\hat{m}, \lfloor d_i \rfloor - 1\}$ greedily from k_1 batteries, of which at least $k_1 - 1$ are emptied. Note that g_i is the remaining current demand from line 15, and that $g_i \geq 1 + w$. Thus, $P_{ALG}(d_i) = \sum_{\{j|x_j>0\}} (1 - x_j) = k_1 - g_i$. We find that $\Delta(i) = k_1 - g_i - w < k_1 - 1$, i.e., $\Delta(i) < e_i$.

If $1 < d_i < m$ and d_i has a high fraction. Let $w = \lceil d_i \rceil - d_i$. It holds that $P_{OPT}(d_i) = w$. The algorithm allocates $g_i = d_i - \min\{\hat{m}, \lfloor d_i \rfloor\}$ greedily from k_1 batteries, of which at least $k_1 - 1$ are emptied. Note that $g_i \geq 1 - w$. Thus, $P_{ALG}(d_i) = \sum_{\{j|x_j>0\}} (1 - x_j) = k_1 - g_i$. We find that $\Delta(i) = k_1 - g_i - w < k_1 - 1$. That is, $\Delta(i) < e_i$.

If $k_2 > 0$ then the condition $k_1 + 2k_2 \geq 3$ might not be valid only if $k_1 = 0$ and $k_2 = 1$. This implies that the whole allocation was from batteries having a remaining capacity between 1 and 1.5, where all but a single battery allocated

exactly 1 and a single battery allocated $1 \leq x_j \leq 1.5$. Such an allocation must be optimal and thus, in this case, $\Delta(i) = 0$. ■

Next, we consider the case in which the batteries' status is not $AboveTheLine$ since $C_m^i < 1$. We bound the redundant penalty in this case and also show that the algorithm might accumulate a redundant penalty at most once before the other condition ($C_1^i < 1.5$) holds.

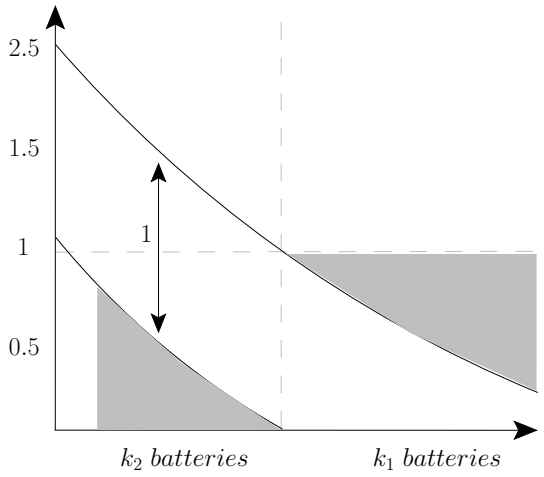

Fig. 3. The batteries status is not $AboveTheLine$ and $C_m^i < 1$.

Claim IV.6. *If $C_m^i < 1$ and* GreedyFinalAllocation *is called, then $\Delta(i) \leq 1.5(e_i + 1)$ and $C_1^{i+1} < 1.5$.*

Proof: By Claim IV.3, if $C_m^i < 1$ then $C_1^i < 2.5$. Assume that *GreedyFinalAllocation* is called. According to the algorithm, this happens only if the allocation is not optimal and only after we set the allocation of each of the $\hat{m}$ batteries with $C_j^i \geq 1$ to 1. Thus, as a result of such an allocation, the remaining capacity of all batteries is less than 1.5, and in particular $C_1^{i+1} < 1.5$. Note that if a low fraction request is serviced and the greedy allocation is of $d_i - (\lfloor d_i \rfloor - 1)$ then $\lfloor d_i \rfloor - 1 < \hat{m}$ and the allocation is non optimal since $d_i < \sum_{j=1}^{\lfloor d \rfloor} C_j$ (see line 18). This can only happen if $C_1^i < 1.5$, thus, the analysis in the proof of Claim IV.5 applies.

Assume that the greedy allocation empties k_1 batteries for which $C_j^i \leq 1$ and k_2 batteries for which $C_j^i > 1$ as shown in Fig. 3. The penalty for the k_1 batteries is $\sum_j (1 - C_j^i) < k_1$. The penalty for the k_2 batteries is $\sum_j (C_j^i - 1) < 1.5k_2$. The last inequality follows from the fact that $C_j^i \leq C_1^i < 2.5$. The total penalty is therefore $P_{ALG}(d_i) < k_1 + 1.5k_2$. All batteries participating in *GreedyFinalAllocation* except perhaps for the last one are emptied, thus, $e_i = k_1 + k_2 - 1$. Clearly, for all values of k_1, k_2, it holds that $k_1 + 1.5k_2 \leq 1.5(k_1 + k_2)$. Thus, $\Delta(i) \leq 1.5(e_i + 1)$. ■

Combining Claims IV.5 and IV.6 we find that for all (except perhaps for one) requests for which $e_i > 0$ it holds that $\Delta(i) \leq 1.5e_i$, and for a single request – the one for which *GreedyFinalAllocation* is called when $C_m^i < 1$, it might be that $1.5e_i \leq \Delta(i) \leq 1.5(e_i + 1)$. To complete our analysis we show that this gap of 1.5 is closed by the last request. In the proof we assume that when the last request is supplied, $C_1^n < 1.5$, as otherwise, the scenario handled in Claim IV.6 never happens and there is no gap to bridge, that is, before the last request $\sum_i \Delta(i) \leq 1.5 \sum_i e_i$.

Claim IV.7. *Either $m' > 0$, or for the last request, $e_i > 0$ and $\Delta(i) \leq 1.5(e_i - 1)$.*

Proof: If all batteries are emptied then the last request empties the last battery. Assume that greedy empties k_1 batteries for which $C_j^i \leq 1$, and k_2 batteries for which $C_j^i > 1$. The penalty for the k_1 batteries is $\sum_j (1 - C_j^i) < k_1$. The penalty for the k_2 batteries is $\sum_j (C_j^i - 1) < 0.5k_2$ (recall that $C_1^n < 1.5$). The total penalty is therefore $P_{ALG}(d_i) < k_1 + 0.5k_2$. All batteries are emptied, thus, $e_i = k_1 + k_2$. We need to show that $k_1 + 0.5k_2 \leq 1.5(k_1 + k_2 - 1)$. The proof is equivalent to the proof of Claim IV.5. ■

We conclude that either the last battery is not emptied, or, if it is emptied, for at least one request (the last one), at least one battery is emptied with no redundant penalty. This compensates for the additional penalty the algorithm might have accumulated in the situation where $C_m^i < 1$ which discussed above. Summing up, the total redundant penalty of the algorithm is at most $1.5m$. ■

V. A Lower Bound for the Multiplicative Competitive Ratio

A more common performance measure of online algorithm's quality is its *competitive ratio*, defined as the maximal possible ratio between the algorithm's objective value and an optimal one. In this section we show that according to this measure no algorithm can be better than 1.5-competitive. Formally,

Theorem V.1. *For every online algorithm, ALG, a sequence of requests σ such that $P_{ALG}(\sigma) \geq 1.5P_{OPT}(\sigma)$ exists.*

Proof: Consider an instance with $m = 2$ batteries with an initial capacity $C = 1$. The sequence σ is constructed by the adversary as a response to the behavior of the algorithm. The possible sequences, provided by the adversary, and the possible allocations of the algorithm are described in Fig. 4. Every node corresponds to a possible configuration of the batteries and is described as a vector of the remaining capacities.

The first request presented by the adversary is for $d_1 = \frac{1}{2} - \varepsilon$. If the algorithm splits the service between the two batteries (configuration A in the figure), then $P_{ALG}(\sigma) = 2 - d_1 = \frac{3}{2} + \varepsilon$, while $P_{OPT}(\sigma) = 1 - d_1 = \frac{1}{2} + \varepsilon$. In this case, the adversary halts; the whole sequence is a single request, and the competitive ratio is arbitrarily close to 3 (by fixing $\varepsilon \to 0$).

If the algorithm supply the whole first request from a single battery, w.l.o.g., from B_1, then the adversary proceeds with $d_2 = \frac{1}{2} - \varepsilon$. The algorithm can choose one of three options: split the service (configuration B), provide the whole request

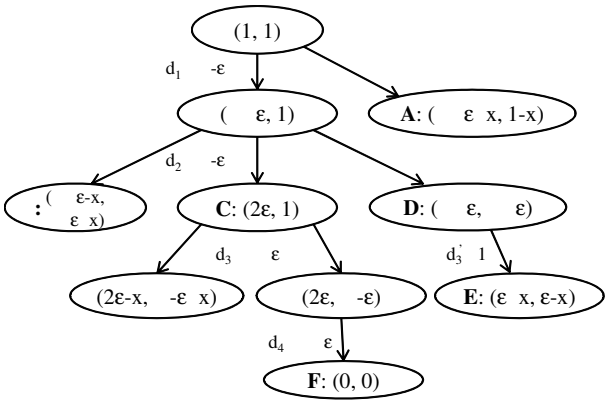

Fig. 4. Possible behavior of an online algorithm and the adversary.

from B_1 (configuration C), or provide the whole request from B_2 (configuration D).

If the algorithm chooses to move to configuration B, the adversary halts. The penalty for the second request is $2-d_2 = \frac{3}{2}+\varepsilon$. In addition to the penalty for the first request, the total penalty is $P_{ALG}(\sigma) = (\frac{1}{2}+\varepsilon)+(\frac{3}{2}+\varepsilon) = 2+2\varepsilon$, while an optimal service has penalty $P_{OPT}(\sigma) = 2(\frac{1}{2}+\varepsilon) = 1+2\varepsilon$. The competitive ratio is arbitrarily close to 2.

If the algorithm chooses to move to configuration D, then the adversary proceeds with $d'_3 = 1$. The algorithm must serve the request d'_3 from both batteries (configuration E) with total penalty for the whole sequence $P_{ALG}(\sigma) = (\frac{1}{2}+\varepsilon)+(\frac{1}{2}+\varepsilon)+1 = 2(1+\varepsilon)$. An optimal service for this sequence (reaching configuration C and providing the whole request d'_3 from B_2) would incur a penalty of $P_{OPT}(\sigma) = 1+2\varepsilon$. The resulting competitive ratio is arbitrarily close to 2.

We are left with the case in which the algorithm chooses to move to configuration C. The adversary proceeds with $d_3 = \frac{1}{2}+\varepsilon$. If the algorithm splits the service (left node of C), the penalty for the last request is $\frac{3}{2}-\varepsilon$ and the total penalty for the whole sequence is $P_{ALG}(\sigma) = \frac{5}{2}+\varepsilon$ while an optimal service for this sequence (providing the whole request d_3 from B_2) would incur a penalty of $P_{OPT}(\sigma) = \frac{3}{2}+\varepsilon$. The resulting competitive ratio is arbitrarily close to $\frac{5}{3}$, and larger than 1.5 for any $\varepsilon \leq \frac{1}{2}$. Otherwise (right node of C), the adversary proceeds with a final request $d_4 = \frac{1}{2}+\varepsilon$. The algorithm must move to configuration F. For the whole sequence the penalty is $P_{ALG}(\sigma) = (\frac{1}{2}+\varepsilon)+(\frac{1}{2}+\varepsilon)+(\frac{1}{2}-\varepsilon)+(\frac{3}{2}-\varepsilon) = 3$, while an optimal service of this sequence (through configuration D) would incur a penalty of $P_{OPT}(\sigma) = 2$. Thus, the competitive ratio for this scenario is 1.5.

We conclude that for any behavior of the algorithm, the adversary can proceed in a way that guarantees $P_{ALG}(\sigma) \geq 1.5P_{OPT}(\sigma)$. ■

VI. Discussion and Open Problems

This paper studied the problem of utilizing the pack of batteries serving current demands in Electric Vehicles. We formulated the problem as a combinatorial optimization problem, provided hardness results, that are valid even for the off-line scenario, and suggested an efficient almost optimal on-line algorithm. Several important problems remain open:

1) Consider additional penalty functions. In particular, non-linear penalty functions as well as penalty functions that are not symmetric around the optimal discharge current.
2) Our algorithm (Section IV) is guaranteed to have redundant penalty at most $1.5m$. On the other hand, it is NP-hard to guarantee redundant penalty lower than $0.5m$, even in the offline case (Theorem III.3). Can this gap be closed? We believe it might be possible to design an algorithm with a lower guaranteed redundant penalty. However, it might also be possible to provide a hardness result with a higher lower bound.
3) Consider the combination of several different battery packs in a single EV, each having its own optimal discharge current.

Additional problems arise when the model is extended to consider additional parameters such as battery temperature and other environmental affects. A totally different direction is to study adaptive switching algorithm, that are based on learning the driver's driving pattern.

References

[1] D. Kirsch, *The electric vehicle and the burden of history*. New Brunswick, NJ: Rutgers University Press, 2000.
[2] C. Anderson and J. Anderson, *Electric and hybrid cars: A history*. McFarland & Company, 2010.
[3] A. Affanni, A. Bellini, G. Franceschini, P. Guglielmi, and C. Tassoni, "Battery choice and management for new-generation electric vehicles," *IEEE Transactions on Industrial Electronics*, vol. 52, no. 5, pp. 1343–1349, 2005.
[4] M. Delucchi and T. Lipman, "An analysis of the retail and lifecycle cost of battery-powered electric vehicles," *Transportation Research Part D: Transport and Environment*, vol. 6, no. 6, pp. 371–404, 2001.
[5] MIT Electric Vehicle Team, "A Guide to Understanding Battery Specifications," http://mit.edu/evt, December 2008.
[6] L. Benini, G. Castelli, A. Macii, and R. Scarsi, "Battery-Driven Dynamic Power Management," *IEEE Design & Test of Computers*, vol. 18, no. 2, pp. 53–60, 2001.
[7] M. Pedram and Q. Wu, "Design Considerations for Battery-Powered Electronics," in *Proceedings of the 36th Annual Conference on Design Automation (DAC'99)*. IEEE Computer Society Washington, DC, USA, 1999, pp. 861–866.
[8] M. Doyle and J. Newman, "Analysis of capacity–rate data for lithium batteries using simplified models of the discharge process," *Journal of Applied Electrochemistry*, vol. 27, no. 7, pp. 846–856, 1997.
[9] F. Laman and K. Brandt, "Effect of discharge current on cycle life of a rechargeable lithium battery," *Journal of power sources*, vol. 24, no. 3, pp. 195–206, 1988.
[10] L. Benini, G. Castelli, A. Macii, E. Macii, M. Poncino, and R. Scarsi, "Extending lifetime of portable systems by battery scheduling," in *Proceedings of the conference on Design, automation and test in Europe*. IEEE Press Piscataway, NJ, USA, 2001, pp. 197–203.
[11] R. Rao, S. Vrudhula, and D. Rakhmatov, "Analysis of discharge techniques for multiple battery systems," in *Proceedings of the 2003 international symposium on Low power electronics and design*. ACM New York, NY, USA, 2003, pp. 44–47.
[12] L. Benini, D. Bruni, A. Macii, E. Macii, and M. Poncino, "Discharge Current Steering for Battery Lifetime Optimization," *IEEE Transactions on Computers*, vol. 52, no. 2, pp. 985–995, 2003.
[13] M. R. Garey and D. S. Johnson, *Computers and Intractability. A Guide to the Theory of NP-Completeness*. New York: W.H. Freeman, 1979.
[14] H. Shachnai, T. Tamir, and O. Yehezkely, "Approximation schemes for packing with item fragmentation," *Theory of Computing Systems*, vol. 43, no. 1, pp. 81–98, 2008.