The Battery Switching Station Scheduling Problem

Tal Raviv

July 2012

Abstract

Battery switching stations (BSS) provide a service that enables extending the traveling range of electric vehicles. Such stations may be the future equivalent of gas stations and they are currently being deployed in several countries. A BSS is a closed loop system that renews its inventory by recharging the batteries. We study the problem of scheduling the charging process in a BSS with the objective of optimizing a weighted measure of service level and cost.

1. Introduction and Problem Definition

In this paper, we introduce a new scheduling and inventory management problem motivated by the business model of Better Place Ltd. (http://www.betterplace.com/). The company sells electric cars with replaceable Lithium Ion batteries and provides battery replacement services in Battery Switching Stations (BSS) scattered around the country. A BSS is equipped with cells where batteries are charged and stored. A robotic arm is used to carry out the battery replacements within a short time. This business model is an attempt to overcome the two main (interrelated) shortcomings of electric cars, namely short travel range and long battery charging times. Should Better Place succeed in penetrating the electric vehicles (EVs) market, it is likely that other car manufacturers will follow their model and that regulatory authorities will step in to impose universal standards for vehicles with replaceable batteries. Hence, in the future, BSSs may become a sustainable substitute for fueling stations.

The successes of EVs is largely depends on the ability of the electricity network and generation facilities to support a significant increase in the consumption. While the electricity networks worldwide are already nearly fully utilized at peak times, there is a significant residual capacity during off peak periods. It is possible to use this capacity to fulfill the additional demand expected from EVs [1]. Diversion of the demand to the of peak hours can be achieved by an appropriate pricing mechanism. Therefore, a new large-scale electricity consumer such as Better Place Ltd. needs to design its operations in such a manner that allows it to exploit the lower electricity fares during off peak hours. Ideally, all the charging effort should be done at the times of the lowest fares (typically during the night). However, such a policy may require a large stock of batteries in the station and this stock may be very expensive to hold. Indeed, in a typical EV, the battery accounts for more than a third of the total cost [2].

The Battery Switching Station Scheduling Problem (BSSSP) is defined as follows: during a time interval $[0, T]$, referred to as the planning horizon, there are I requests for battery replacements. The BSS must fulfill these requests. Serving a request involves dismantling an empty battery[1] from the car and installing a charged one instead. This operation is assumed to be instantaneous. Later on, the dismantled batteries can be fully recharged in C time units and used to fulfill future requests. Each request i, is characterized by its occurrence time t_i. The BSS is a closed loop system; hence, the total number of batteries in the stations is constant. We denote this number by N.

The supplied battery is preferably fully charged, but a partly charged battery can be provided at a penalty cost that depends on the charging level of the battery. This level is measured by the total charging time that the battery gained. The penalty incurred by a provision of a partially charged battery is a decreasing function $\pi: [0, C] \to \mathbb{R}_+$ with $\pi(C) = 0$. It is reasonable to assume that $\pi(x)$ is convex since this results in a fair allocation of the charging effort among customers when the total penalty is minimized, e.g., the system is better off with two customers who obtained a 50% charge battery compared to one customer that obtains an empty battery and one that obtains a fully charged battery.

Throughout this paper we assume that the penalty function is continuous k-piecewise linear. While this assumption is not necessarily accurate, it provides sufficient flexibility to represent approximately any reasonable penalty function. We explicitly express it as

$$\pi(x) = \begin{cases} \alpha_1 - \beta_1 x & 0 = r_0 \le x \le r_1 \\ \alpha_2 - \beta_2 x & r_1 \le x \le r_2 \\ \vdots & \vdots \\ \alpha_k - \beta_k x & r_{k-1} \le x \le C = r_k \end{cases}$$

Where convexity implies $\beta_1 \ge \beta_2 \ge \cdots \ge \beta_k$, continuity implies $\alpha_p - \beta_p r_i = \alpha_{p+1} - \beta_{p+1} r_i$. Also, $\pi(C) = 0$ implies $\alpha_k = C\beta_k$. For the computational complexity analysis below we assume that constants $\alpha_1, \dots., \alpha_k, \beta_1, \dots., \beta_k$ and $r_1, \dots, r_{k-1}$ are part of the input.

We assume without loss of generality that the charging process consumes one unit of electrical energy per time unit. During the charging process, energy is consumed and accumulated in the battery at a constant rate[2]. The maximal number of batteries that can be charged simultaneously at time t (referred to as the *station capacity*) is denoted by the step function $M(t)$. Clearly, $M(t) \le N$; in some cases the inequality holds strictly due to limitations of the electricity grid. The electricity price varies over time. The price for time t is denoted by the step function $E(t)$. Since the charging process is controlled electronically and the batteries are charged in their cabinets, preemptions can be carried out at no cost. The Lithium Ion battery technology allows resuming the charging process from the same point later on. The initial charging level of the batteries at the beginning of the planning horizon is denoted by $(v_1, v_2, \dots, v_N)$.The goal is to schedule the charging process of the batteries in the system and to decide which battery should be used to meet each request so as to minimize total electricity and penalty costs.

In practice, the demand for batteries is stochastic rather than deterministic. That is, the amount and timing of requests is unknown in advance and an online algorithm that responds to

[1] In practice, the batteries dismantled from the cars are not completely empty. We shall revisit this simplifying assumption in Section 3.

[2] This is not necessarily the case for the charging process of Lithium Ion batteries. We shall revisit this simplifying assumption in Section 3.

the actual demand is needed. One possible strategy toward this online problem is to solve the deterministic model repeatedly based on the actual state of the system and the forecasted demand and to role the horizon after each point of demand. Indeed, Einy [3] shows that this is a successful heuristic when applied with a reasonable level of safety stock. Our main contribution in this paper is in presenting an efficient algorithm that allows solving the deterministic sub-problem of such a heuristic quickly enough to be used on-line.

The BSSSP is closely related to previously studied scheduling problems of parallel machines with preemption, release times and due dates. See for example Lawler and Labetoulle [4], Gonzalez and Sahni [5], Federgruen and Groenevelt [6], and Sheen and Liao [7]. However, the BSSSP is complicated by the fact that release times are not specified; instead, the planner needs to decide which battery should be used to serve each future request. The variable electricity costs and the opportunity to supply semi-finished jobs (i.e., partially charged batteries) further enrich this scheduling model.

The rest of this paper is organized as follows: in Section 2, we present a strongly polynomial time algorithm for the BSSSP. In Section 3, we remove some of the simplifying assumptions and introduce an extended version of the problem. We present an effective Mixed Integer Formulation for the problem and identify conditions under which the problem is solvable in pseudo polynomial time. In Section 4, we present the result of a numerical experiment with our formulation, using data inspired from an actual BSS, and demonstrate its suitability for an online setting. We conclude with some remarks and directions for further research in Section 5.

2. Polynomial Time Algorithm for the BSSSP

In this section, we present a strongly polynomial time algorithm for the BSSSP as defined above. We first identify useful properties of some optimal solutions of the problem. For a given schedule and a request, let us define the *occupancy interval* of a request (for battery replacement) as the time interval in which the system carries the battery that fulfills the request. We denote the starting time of the occupancy interval of request i by s_i. Obviously, the interval ends when the request is fulfilled at time t_i. If request i is fulfilled with a battery that was initially available in the system, then $s_i = 0$. The starting of the occupancy intervals of the rest of the requests must be equal to the time t_j, of some previous request j (where $j < i$). Each request time is associated with at most one starting time of an occupancy interval of a subsequent request.

Let $Q_i \subset [s_i, t_i]$ be the set of time intervals in which the battery used to fulfill request i was actually being charged. Since preemptions in the charging process are allowed, Q_i is not necessarily a single continuous interval. Note that the values of $Q_1 Q_2, \dots, Q_I$ together with the identity of the battery used to fulfill each request uniquely define a solution of the BSSSP. In this case, identity is designated by the arrival time of the battery to the system or in the case of a battery that is available at time 0, its initial charging level. A schedule is said to be feasible if the number of batteries that are charged simultaneously never exceeds the charging capacity of the station. That is, $\sum_i \mathbb{I}_{\{t \in Q_i\}} \leq M(t)$ for all $t \in [0, T]$, where $\mathbb{I}_{\{\cdot\}}$ is an indicator function.

We define the *FIFO fulfillment policy* as follows: whenever a request is received in the system, it is fulfilled by a battery whose arrival time at the system is the oldest or in the case of batteries with $s_i = 0$, by the battery with the highest initial charging level. Ties are broken in favor of batteries with the smallest remaining charging time at the time of the request. This policy can be described alternatively as follows: the first N requests are fulfilled by the N batteries that were already in the system at the beginning of the planning horizon in non-increasing order of their initial charging levels, v_i. For $i = N+1, \dots, I$, the i^{th} request is served using a battery that was returned by the $(i-N)^{th}$ request. That is,

$$s_i = \begin{cases} 0 & i \le N \\ t_{i-N} & i > N \end{cases}$$

Proposition 1: If the penalty for fulfillment of a request with partially charged batteries is a convex function of the charging level, an optimal schedule that admits the FIFO fulfillment policy always exists.

Proof: We prove the proposition by contradiction, showing that any feasible solution of the problem can be modified to one that admits FIFO fulfillment policy while satisfying the capacity constraint with smaller or equal electricity and penalty costs. Consider a feasible schedule, and let requests i and j be the first pair of requests that violate the FIFO policy. That is, in this schedule $s_i < s_j$ but $t_i > t_j$. Later on in this proof, we fix the values of s_i and s_j even though their meaning is swapped.

Let us denote the set of times allocated for charging the battery that fulfills request i during the interval $[s_i, s_j)$ by A_1, that is $A_1 \equiv Q_i \cap [s_i, s_j)$. Similarly $A_2 \equiv Q_i \cap [s_j, t_j)$ and $A_3 \equiv Q_i \cap [t_j, t_i)$. See Figure 1 for an example.

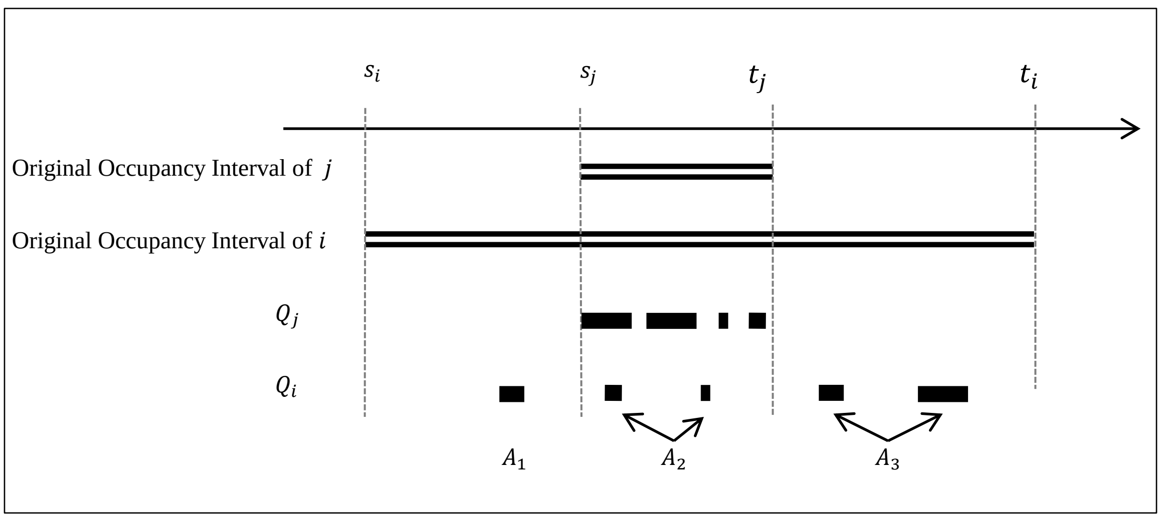

Figure 1: Original schedule (Case 1)

We consider two cases, namely, Case 1: $|Q_j \setminus A_2| \ge |A_1|$ and Case 2, otherwise. In Case 1, let B be a subset of $Q_j \setminus A_2$ such that $|B| = |A_1|$. In such a case, it is possible to modify the occupancy interval of request j to $[s_i, t_j]$, the charging periods to $Q'_j = A_1 \cup \{Q_j \setminus B\}$ and reschedule the charging of the battery that fulfill request i such that the occupancy interval is $[s_j, t_i]$ and charging time $Q_i' = A_2 \cup A_3 \cup B$. See Figure 2.

Note that in this case the total charging time allocated to the batteries that were provided for each of the requests i and j as well as all other requests, was left unchanged so there is no change in the penalty cost. In addition, the total electricity consumption at each moment was left unchanged and thus the total electricity cost is unaffected.

In Case 2 (see for example Figure 3), when $|Q_j \setminus A_2| < |A_1|$ it must be the case that $|Q_j| \le |A_1| + |A_2|$ and therefore $|Q_j| \le |Q_i|$. It is possible to reschedule the charging of the batteries that fulfill the two requests as follow. $Q_j' = A_1 \cup A_2$ and $Q_i' = Q_j \cup A_3$. See Figure 4. Again, in this new feasible schedule, the total electricity consumption at each moment as well as the total charging efforts allocated to both batteries, are left unchanged. However, the total charging time allocated for request j is increased and the total charging time allocated for request i is decreased by the same amount. Now the total penalty incurred by both batteries is

$$\pi(|Q_i'|) + \pi(|Q_j'|) = \pi(|Q_j| + |A_3|) + \pi(|A_1| + |A_2|) \le p(|Q_j|) + \pi(|A_1| + |A_2| + |A_3|)$$
$$= \pi(|Q_i|) + \pi(|Q_j|)$$

The inequality is due to the convexity of $\pi(\cdot)$ and the fact that $|Q_j| \le |Q_j'|, |Q_i'| \le |Q_i|$. Intuitively speaking, dividing the charging effort between the two requests more evenly reduced the total penalty. ■

An important consequence of Proposition 1 is that it is possible to determine the occupancy intervals of all the requests in an optimal solution merely by sorting the requests according to their times.

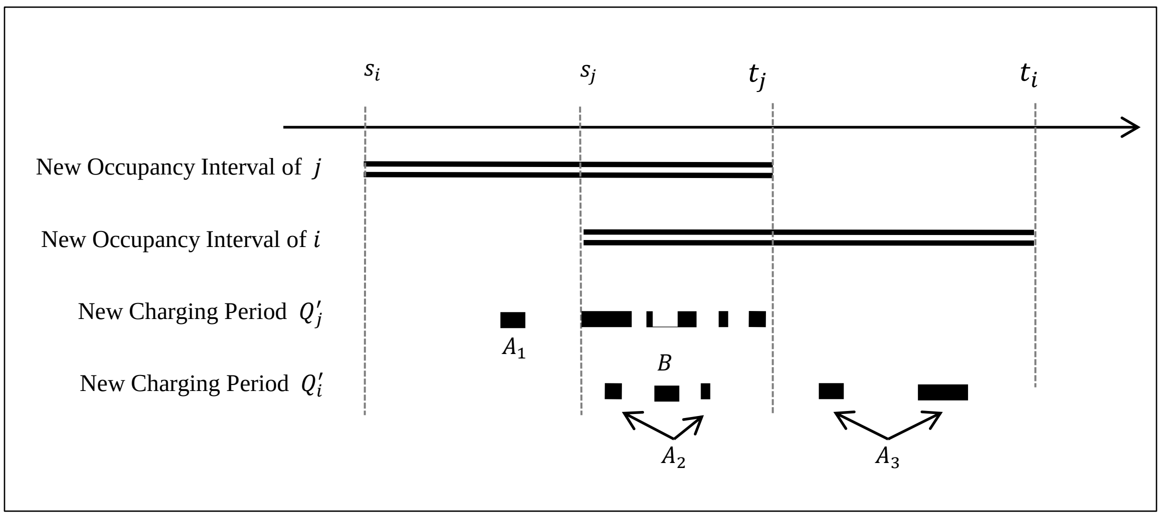

Figure 2: Modification of the schedule (Case 1)

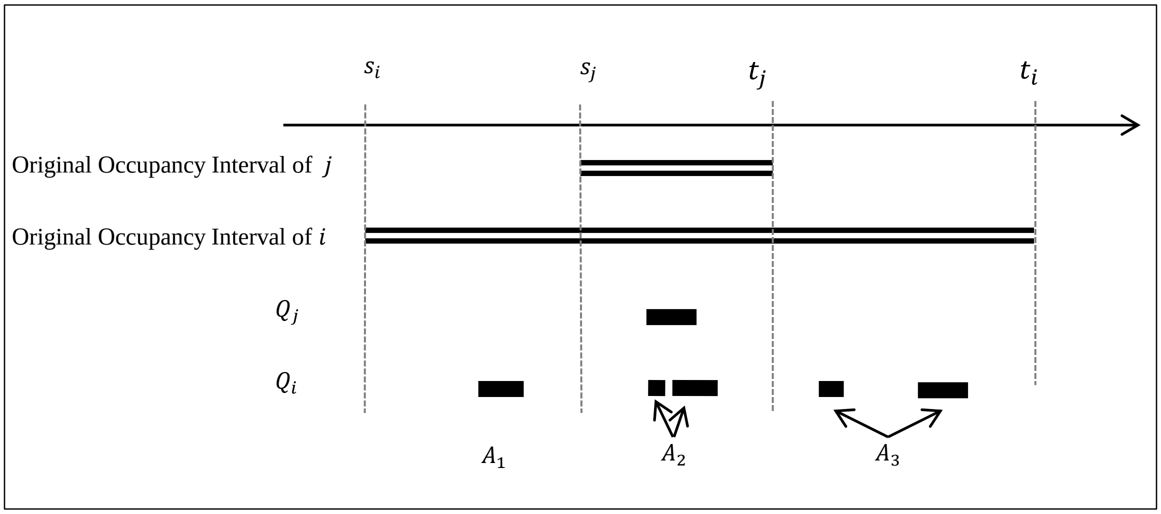

Figure 3: Original schedule (Case 2)

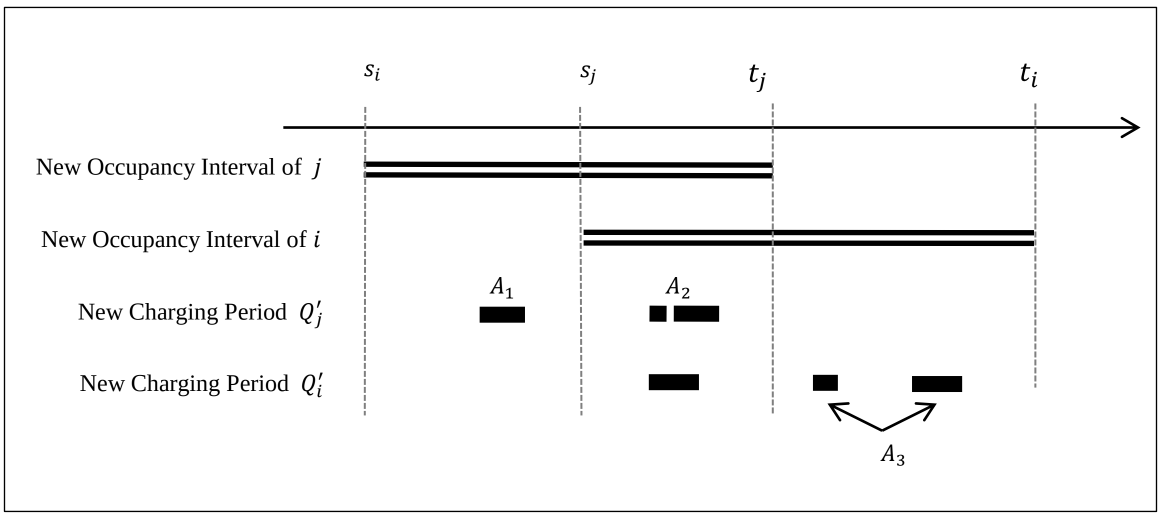

Figure 4: Modification of the schedule (Case 2)

Let us define an *event* in the system as a point in time in which one or more of the following occurs: a request is accepted, the electricity cost is changed, or the charging capacity of the system is changed. The planning horizon is divided into *epochs*, where each event marks the end of one epoch and the beginning of the next. The first epoch starts at time 0. Clearly, each occupancy interval is a union of several epochs. Let Q_i denote the set of epochs that constitute the occupancy interval of request i.

Based on the result of Proposition 1 we can construct an optimal solution for the BSSSP using a three steps algorithm as follows:

Step 1: Construct the sets of epochs $Q_1, Q_2, \dots, Q_I$ for all requests according to FIFO fulfillment policy.

Step 2: Decide upon the charging time allocated to each request during each epoch so as to minimize total electricity and penalty costs while satisfying the charging capacity constraint. That is, the total charging time during the epoch is no greater than the epoch length multiplied by the charging capacity available during the epoch.

Step 3: Schedule the charging intervals of each request within each epoch, such that the capacity constraint is satisfied at throughout during the epoch.

The implementation of step 1 follows directly from the definition of the FIFO policy and the epochs above. Step 2 can be accomplished by a reduction of the problem to a minimum cost flow problem, as described below. Solving the problems of step 3 for each epoch is equivalent to finding a feasible preemptive schedule for a set of jobs on identical parallel machines given a common due-date, $P|prmpt, d_i = d|feasibility$; a problem that can be solved in linear time by a greedy procedure, see [8].

Let us denote the length of each epoch e by l_e, the charging capacity during the epoch by $\mathcal{M}_e$, and the electricity price by $\mathcal{E}_e$. In order to solve the problem of step 2, let us construct a network with a node e for each epoch; a node i for each request, one sink node, k penalty nodes - one for each piece of the piecewise linear penalty function. Let us index these nodes by p. The supply at each event node e is $l_e \times \mathcal{M}_e$. This represents the total charging time that can be allocated during the epoch to all the requests. The supply of each penalty node p is $I\left(r_p - r_{p-1}\right)$. The demand (negative supply) of each request node $i \leq N$ is $C - v_i$ and the demand of the rest of the request nodes is C. This represents the total charging time required by each request. Next, we create arcs from each epoch node e to each request node i that satisfies $e \subseteq Q_i$. That is, the occupancy interval of the request covers the epoch. The capacity of each arc (e, i) is l_e and its cost per unit of flow is $\mathcal{E}_e$. In addition, each penalty node p is connected to each request node by an arc of capacity $r_p - r_{p-1}$ and cost β_p. Finally, each event node and each penalty node is connected by an uncapacitated zero cost arc to the sink. The demand of the sink node is

$$IC + \sum_e l_e \times M_e - \sum_{i=1}^{N} v_i - (I - N) \times C = NC + \sum_e l_e \times M_e - \sum_{i=1}^{N} v_i$$

so it can absorb all the residual supply. An optimal solution of the minimum cost flow problem for this network prescribes an optimal solution for step 2. The flow on arc (e, i) represents the total time allocated during the epoch e, for charging the battery that fulfills request i.

Let us demonstrate the algorithm using a small numerical example. Consider an instance of the problem with: $I = 4, C = 4, N = 2, v_1 = 2, v_2 = 3, M(t) = 2$ for all t, $E(t) = 1$ for $t \leq 4$ and $E(t) = 2$ for $t > 4$. The request times are $(t_1, t_2, t_3, t_4) = (3,5,7,8)$. Let the penalty function be

$$\pi(x) = \begin{cases} 30 - 10x & 0 \leq x \leq 2 \\ 20 - 5x & 2 \leq x \leq 4 \end{cases}.$$

That is $r_0 = 0, r_1 = 2, r_2 = 4$ and $\beta_1 = 10, \beta = 5$.

It easy to check that the starting times of the occupancy intervals according to FIFO are $(s_1, s_2, s_3, s_4) = (0,0,3,5)$. There are five epochs, namely [0,3], [3,4], [4,5], [5,7], [7,8]. In Figure 5, we present the resulting minimum cost flow problem with the supply (and demand as negative supply) shown next to each node and with the capacities and the costs of the flows presented in parenthesis above some of the arcs or at the legend at the bottom right corner of the figure.

An example of a flow in this network is presented in Figure 6 where only the arcs with positive flow are drawn. The flow values are presented near the arcs. The width of the arcs is proportional to the cost of flow per unit in each arc. Thin dashed lines represent arcs with zero cost. The flow represents a solution in which request 1 is provided by a battery with initial charging level of 2 that is charged for some two units of time, during the interval [0,3] at electricity cost of 2. Request 2 is charged during the entire interval [0,3] at electricity cost of 3. Request 3 is charged during the period [3,7] at electricity cost of $1 + 2 \times 3 = 7$. Request 4 is charged during the period [5,8] at electricity cost of 6. In addition, since the battery is provided with only 3 units of energy (out of 4). A penalty of 5 is charged. The total electricity and charging cost is $2 + 3 + 7 + 6 + 5 = 23$. It can be verified that this is equal to the total flow cost depicted in Figure 6.

Note that the solution of Step 2 does not prescribe a concrete charging schedule. For example, the battery of request 1 is charged for some two time units during the first epoch that spans over the time interval [0,3] but we still do not know what is the exact timing of the charging operation. However, since the total time of the charging operations in the first epoch is $2 + 3 = 5$ and the charging capacity during this epoch is 2, it is always possible to schedule the operations (possibly with preemption) in a greedy manner. In this case, for example one can schedule the first request at time 0 and once it is completed at time 2, start with the second request, continue until the end of the epoch at time 3, preempt it and start again on the second capacity unit at time 0 until time 2. The capacity constraints on the arcs guarantee that such a schedule will be feasible at each epoch.

Let k denote the number of events, proportional to the size of the input of the problem. The number of nodes in the constructed bipartite network is $O(k)$ and the number of arcs is $O(k^2)$. Orlin [9] introduced the fastest algorithm yet for the minimum cost flow problem with a complexity of $O\big(m\, log\, n(m + n\, log\, n)\big)$ for networks with n nodes and m arcs. This leads to a strongly polynomial time algorithm for the BSSSP with complexity of $O\big(k^2\, log\, k(k^2 + k\, log\, k)\big) = O(k^4 \log k)$. The complexities of steps 1 and 3 are clearly dominated by the complexity of step 2.

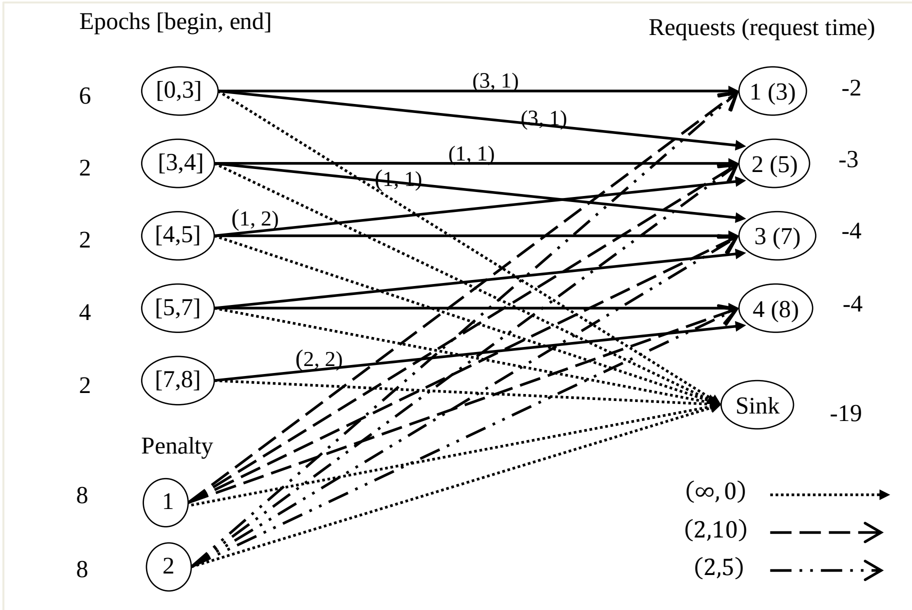

Figure 5: Example of the reduction of Step 2 into minimum cost flow problem

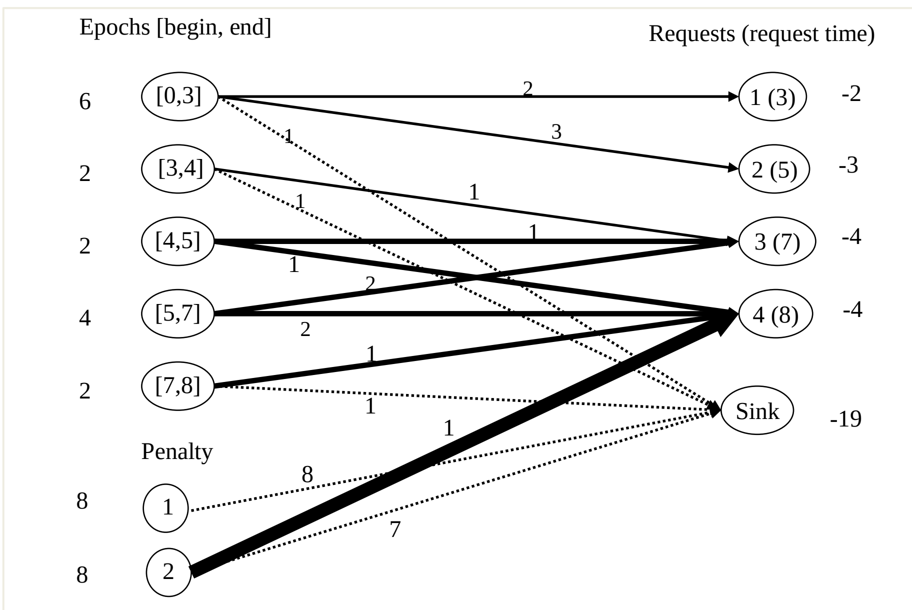

Figure 6: Feasible solution for the min cost flow problem presented in Figure 5

3. Extended BSSSP

In this section, we present and discuss an extended version of the BSSSP model that removes two simplifying assumptions of the original model. We refer to this model as EBSSSP. In particular, we no longer assume that all the batteries are returned completely empty from the customer, and that the energy consumption is constant during the charging process of the batteries. We note that the FIFO policy that we proved to be optimal for BSSSP is no longer optimal for this extended model; hence the problem may be essentially harder.

The *charging phase* (hereafter, *phase*) of a battery represents the charging time equivalent energy that it contains. For examples, the phase of a completely charged battery is C and the phase of a battery that requires additional x time units to be charged completely is $C - x$.

We formally describe the EBSSSP with an integer linear program (ILP) that assumes discretized time information. That is, we assume that all the events occur at discrete time units and that the residual charging time of the returned batteries is discrete. If the time units are fine enough, the solution of such a model can be applied with adequate accuracy to the actual continuous time problem. First, we present some additional notations to be used by the mathematical model

d_k^t The number of requests at time t of customers that return batteries at phase k

R_k The power required to charge a phase k battery (until it reaches phase $k+1$) in terms of Watts.

M_t The station capacity at time t, as before, but now in terms of available electrical power (Watts)

Our model uses the following decision variables

x_k^t The number of batteries at phase k that are being charged during the interval $[t, t+1)$. The values of x_k^0 may be constants that represents decisions made in the last period before the planning horizon (e.g., in a rolling horizon setting) or can be omitted from the model if the system starts at period 1.

y_k^t The number of batteries at phase k that are supplied to customers at time t

b_k^t The number of batteries at phase k in the station at time t just after the provision of batteries to the customers is carried out, at the beginning of the period. The value of this variable does not include batteries that are being charged during the period $[t, t+1)$. The values of b_k^0 are constants that represent the initial inventory.

The problem can be now stated as,

$$\min \sum_{t=1,k=0}^{T,C-1} E_t R_k x_k^t + \sum_{t=1,k=0}^{T,C-1} \pi(k) y_k^t \tag{1}$$

Subject to

$$\sum_{k=0}^{C-1} R_k x_k^t \le M_t \qquad \forall t = 1, \dots, T \tag{2}$$

$$b_k^t = b_k^{t-1} + x_{k-1}^{t-1} - x_k^t - y_k^t + d_k^t \qquad \forall t = 1, \dots, T, k = 0, \dots, C \tag{3}$$

$$\sum_{k=0}^{C} y_k^t = \sum_{k=0}^{C-1} d_k^t \qquad \forall t = 1, \dots, T \tag{4}$$

$$x_k^t \ge 0, integer \qquad \forall t = 1, \dots, T, k = 0, \dots, C-1 \tag{5}$$

$$y_k^t, b_k^t \ge 0, integer \qquad \forall t = 1, \dots, T, k = 0, \dots, C \tag{6}$$

The objective function (1) is the sum of electricity costs and the penalty costs due to delivery of partially charged batteries. Constraints (2) are capacity constraints to make sure that the electrical power consumed at each moment does not exceed the capacity of the station. Constraint (3) is an inventory balance constraint that relates the inventory level of the batteries at each phase with the inventory level at the previous period, the number of batteries supplied to customers, the number of batteries returned by customers and the charging decisions. Constraints (4) stipulate that the number of batteries provided to the customers at each period is equal to the number of requests. Constraints (5) and (6) require integrality and non-negativity of all the decision variables. Note that x_k^t is defined only for $k = 1, \dots, C - 1$. Therefore, constraints (3) omit the relevant terms at the boundaries.

Unfortunately, we do not know what the complexity status of the extended problem is. It is impossible to derive conclusions about this question from the above MIP. However, our numerical experiment in Section 5 shows that it is possible to solve the model using a commercial solver within a short time and with satisfactory numerical accuracy (i.e., time units of five minutes).

Next, consider the *uncapacitated* case of the problem where the electricity power at the station suffices to charge all the batteries simultaneously[3]. In terms of MIP (1)-(6), this special case is obtained by removing the capacity constraint (2). Note that without this constraint the MIP is reduced to a network flow model that can be solved in polynomial time with respect to the size of the input. This implies a pseudo polynomial algorithm for the uncapacitated EBSSSP since the number of nodes and arcs in this network depends on T and C.

4. Numerical Experiment

In order to test the effectiveness of the time indexed MILP formulation presented above we constructed a set of 72 benchmark instances. To this end we generated six sets of requests that occurred over a planning horizon of $T = 1200$ minutes (20 hours) starting at 4:00 am and ending at midnight. The characteristic of these demand realizations are detailed in Table 1.

Name	Number of requests	Description
D1	1000	Morning and afternoon demand peaks, typical of regular working days.
D2	1500	
D3	2000	
D4	1000	Single demand peak around noon time, typical of Saturdays (in Israel)
D5	1500	
D6	1500	Requests are randomly spread along the planning horizon

Table 1: Description of the six demand realizations used in the numerical experiment

We created instances with two different electricity tariff patterns offered by the Israel Electric Company for high voltage business Clients during three different seasons on regular working days. Namely: Summer Tariff, Winter Tariff (see

[3] From private communication with Better Place Ltd. personal, we learnt that this is currently the situation in most of the stations.

http://www.iec.co.il/BusinessClients/Pages/UpperVoltage.aspx). Note that the "fall and spring" tariff currently offered by the company exhibits little variations throughout the day and thus has little opportunity for savings using a smart charging schedule. The total amount of energy required to fully charge a battery is about 30KWh. The process requires one hour[4] ($C = 60\text{min.}$). The energy consumption rate during the charging process was assumed constant, which is approximately the case with the fast charging technology currently used by Better Place Ltd. Therefore, the power consumption during the charging process of a battery is 0.5KWh per minute. The charging phase of the returned batteries in our experiment was drawn from normal distribution with a mean of 15 minutes and standard deviation of 4 minutes, rounded to nearest integer and truncated at zero.

We note that since the electricity price during the night is constant and minimal, it is always optimal to charge at full capacity during these hours. Moreover, since the expected demand at these hours is very small, the order in which the batteries are charged is unimportant. Therefore, we decided to omit these hours from the experiment. For this reason, we also assume that the initial state of all the batteries in the station, at 4:00 am, is "fully charged".

We tested three battery inventory levels at the stations, namely, $N = 100, 150, 200$. The charging capacity of the station in our test is constant at either 50% or 75% of the number of slots. That is $M_t = 50{,}75$ for $N = 100$; $M_t = 75, 113$ for $N = 150$; and $M_t = 100{,}150$ for $N = 200$. The penalty function was set to $\pi(x) = \left((C - x)/5\right)^2$ where the argument, x, is the phase of the supplied battery in minutes. Recall that this function is approximated by a piecewise linear function in the model where the number of pieces equal to the number of periods in C.

Overall, we created instances with 6 demand patterns × 2 electricity tariffs × 3 station capacities × 2 charging capacities = 72 problem instances. The request times and remaining charging times are given in discrete number of minutes but a preliminary experimentation with this level of discretization indicates that the solution times at this level are too large. Recall that in practice the problem needs to be solved on-line with a rolling horizon in order make scheduling decisions in a stochastic environment. Hence, we decided to discretize the time to five minutes units. In these terms the time dimension of the benchmark problems are $T = 240, C = 12$. For a comprehensive report on the application of a similar deterministic model in realistic settings within a rolling horizon framework, see [3].

Table 2 presents the optimal cost (in Shekels) and running time in seconds for the 72 instances described above. The model was solved using IBM Ilog Cplex 12.2 with defualt settings on an Intel Core i7 (870) desktop.

Several interesting observations can be made based on these results. First, it should be noted that all the 72 instances could be solved very quickly with an average solution time of 1.27 seconds and about six seconds for the hardest instances. This implies that the integer programming formulaion with time descretization of five minutes and rolling horizon of twenty hours can be used to make scheduling decisions online in a BSS. It is also worth mentioning that 62 out of the 72 instances were solved at the root node of the branch and bound tree, which implies that the time index formulation is pretty tight.

It is clear that a longer horizon and finer time discretization will result in longer running times. Therefore, we checked the effect of other dimensions of the problem on the solution time.

[4] We note that it is technically possible to charge the batteries at various rates. The tradeoff between quick vs. slow charging rates is out of the scope of this paper.

To this end, we ran a linear regression on the 72 observations defining the solution time as a dependent variable. The independent variables were $\mathrm{M}, \mathrm{N}, \mathrm{I}$ and the ratio between number of slots and charging capacity N/M. It turned out that none of the independent variables except for N/M is significant at $\alpha = 0.01$ and the ajusted R^2 of the regression was small ($\overline{\mathrm{R}}^2 = 0.265$). This analysis implies that the effects of the number of requests, the number of slots and the charging capacity of the station on the solution time of our MILP formulation is negligible. Hence, the same solution approach is likely to scale up for larger and busier BSSs then those used in our benchmark instances.

It is apparent from Table 2 that the optimal total cost is much higher for the summer electricity instances compared to those of the winter. This can be attributed to the fact that in the summer the daytime electricity tariff in Israel is higher as the demand soars due to the usage of air conditioners. Therefore, during the summer, the peak demand for batteries replacement coincides with the peak electricity prices hours. Finally, as expected, the total electricity and penalty cost at each demand level is reduced as the number of batteries and the charging capacity are increased.

All the input of our experiment and an internally documented IBM Ilog Opl models can be obtained from the author upon request.

N	*M*	Demand	I	Winter Prices		Summer Prices	
				Cost	Time	Cost	Time
100	50	D1	1000	8681.6258	1.592	11749.901	0.546
100	75	D1	1000	8465.0633	0.558	11649.013	0.445
150	75	D1	1000	7322.3521	0.823	10540.92	0.37
150	113	D1	1000	7322.3521	0.384	10540.92	0.343
200	100	D1	1000	6394.8931	0.382	9433.8702	0.434
200	150	D1	1000	6394.8931	0.337	9433.8702	0.349
100	50	D2	1500	21309.492	1.275	24296.172	3.278
100	75	D2	1500	14974.286	0.565	19451.455	0.519
150	75	D2	1500	12739.294	6.004	17795.133	0.786
150	113	D2	1500	12240.263	1.123	17178.881	0.633
200	100	D2	1500	11051.505	1.893	16171.525	0.462
200	150	D2	1500	11026.562	0.485	16063.747	0.387
100	50	D3	2000	21309.492	1.258	24296.172	3.232
100	75	D3	2000	14974.286	0.535	19451.455	0.536
150	75	D3	2000	12739.294	5.823	17795.133	0.786
150	113	D3	2000	12240.263	1.053	17178.881	0.626
200	100	D3	2000	11051.505	1.811	16171.525	0.457
200	150	D3	2000	11026.562	0.459	16063.747	0.387
100	50	D4	1000	21309.492	1.255	24296.172	3.231
100	75	D4	1000	14974.286	0.563	19451.455	0.517
150	75	D4	1000	12739.294	5.785	17795.133	0.791
150	113	D4	1000	12240.263	1.028	17178.881	0.619
200	100	D4	1000	11051.505	1.841	16171.525	0.445
200	150	D4	1000	11026.562	0.472	16063.747	0.372
100	50	D5	1500	21309.492	1.258	24296.172	3.225
100	75	D5	1500	14974.286	0.554	19451.455	0.525
150	75	D5	1500	12739.294	5.842	17795.133	0.77
150	113	D5	1500	12240.263	1.03	17178.881	0.622
200	100	D5	1500	11051.505	1.814	16171.525	0.446
200	150	D5	1500	11026.562	0.483	16063.747	0.386
100	50	D6	1500	21309.492	1.258	24296.172	3.228
100	75	D6	1500	14974.286	0.544	19451.455	0.512
150	75	D6	1500	12739.294	5.777	17795.133	0.821
150	113	D6	1500	12240.263	1.01	17178.881	0.637
200	100	D6	1500	11051.505	1.82	16171.525	0.45
200	150	D6	1500	11026.562	0.462	16063.747	0.376

Table 2: Summary of the results of the numerical experiment

5. Conclusions and Further Research

This paper presents a mathematical model and effective solution method for a timely practical scheduling problem originating from an emerging technology of electric cars with replaceable batteries. The model, with simple modifications can be applied to various scheduling problems in environments with time varying production costs, e.g., due to a smart electricity grid.

A strongly polynomial time algorithm for a simplified version of the problem (BSSSP) is presented. This version of the problem assumes that batteries are returned completely depleted and that the charging process consumes constant power. Both assumptions can be viewed as conservative assumptions that may provide adequate slack for implementation in a stochastic environment.

The EBSSSP removes the above simplifying assumptions. We present an effective MIP formulation for this problem as well as a pseudo polynomial algorithm for its uncapacitated version. The ability of a commercial solver to deliver optimal solutions for large-scale instances of the problem within a few seconds using this formulation was demonstrated. Thus, it is sufficiently effective to be used as a subroutine of an on-line math heuristic solution approach for the underlying stochastic and dynamic problem. Such a heuristic is currently being devised in a thesis [3] and will be reported upon in a subsequent paper. Inquiry into the complexity status of EBSSSP was left for future research but we believe that the general problem is strongly NP-Hard.

References

[1] M.W. Max Grünig, Benjamin Boteler, Ravi Kantamaneni, Etienne Gabel, Dorien Bennink, Huib van Essen, Bettina Kampman, Impacts of Electric Vehicles – Deliverable 3, Assessment of the future electricity sector, 2011.

[2] J. Goods, Green Update: Why Do EVs Cost So Much?, 2012.

[3] G. Einy, Stochastic Lot Sizing with Variable Production Costs: an Application for Battery Switching Stations, Industrial Engineering, Tel Aviv University, 2012.

[4] E.L. Lawler, J. Labetoulle, Preemptive Scheduling of Unrelated Parallel Processors by Linear-Programming, J Acm, 25 (1978) 612-619.

[5] T. Gonzalez, S. Sahni, Preemptive Scheduling of Uniform Processor Systems, J Acm, 25 (1978) 92-101.

[6] A. Federgruen, H. Groenevelt, Preemptive Scheduling of Uniform Machines by Ordinary Network Flow Techniques, Manage Sci, 32 (1986) 341-349.

[7] G.J. Sheen, L.W. Liao, Scheduling machine-dependent jobs to minimize lateness on machines with identical speed under availability constraints, Comput Oper Res, 34 (2007) 2266-2278.

[8] S. Sahni, Preemptive Scheduling with Due Dates, Oper Res, 27 (1979) 925-934.

[9] J.B. Orlin, A Faster Strongly Polynomial Minimum Cost Flow Algorithm, Oper Res, 41 (1993) 338-350.