Peak Shaving Through Resource Buffering*

Amotz Bar-Noy Matthew P. Johnson Ou Liu

The Graduate Center of the City University of New York

Abstract. We introduce and solve a new problem inspired by energy pricing schemes in which a client is billed for peak usage. At each timeslot the system meets an energy demand through a combination of a new request, an unreliable amount of *free source* energy (e.g. solar or wind power), and previously received energy. The added piece of infrastructure is the *battery*, which can store surplus energy for future use. More generally, the demands could represent required amounts of energy, water, or any other *tenable* resource which can be obtained in advance and held until needed. In a feasible solution, each demand must be supplied on time, through a combination of newly requested energy, energy withdrawn from the battery, and free source. The goal is to minimize the maximum request. In the online version of this problem, the algorithm must determine each request without knowledge of future demands or free source availability, with the goal of maximizing the amount by which the peak is reduced. We give efficient optimal algorithms for the offline problem, with and without a bounded battery. We also show how to find the optimal offline battery size, given the requirement that the final battery level equals the initial battery level. Finally, we give efficient H_n-competitive algorithms assuming the peak *effective* demand is revealed in advance, and provide matching lower bounds.

1 Introduction

There is increasing interest in saving fuel costs by use of renewable energy sources such as wind and solar power. Although such sources are highly desirable, and the power they provide is in a sense free, the typical disadvantage is unreliability: availability depends e.g. on weather conditions (it is not "dispatchable" on demand). Many companies seek to build efficient systems to gather such energy when available and store it, perhaps in modified form, for future use [16].

On the other hand, power companies charge some high-consumption clients not just for the total amount of power consumed, but also for how quickly they consume it. Within the billing period (typically a month), the client is charged for the amount of energy used (*usage charge*, in kWh) and for the maximum amount requested over time (*peak charge*, in kW). If demands are given as a sequence $(d_1, d_2, \ldots, d_n)$, then the total bill is of the form $c_1 \sum_i d_i + c_2 \max_i\{d_i\}$ (for some constants $c_1, c_2 > 0$), i.e., a weighted sum of the total usage and the maximum usage. (In practice, the discrete timeslots may be 30-minute averages [2].) This means that a client who powers a

* This work was supported by grants from the National Science Foundation and the New York State Office of Science, Technology and Academic Research.

100kW piece of machinery for one hour and then uses no more energy for the rest of the month would be charged more than a client who uses a total of 100kWh spread evenly over the course of the month. Since the per-unit cost for peak charges may be on the order of 100 times the per-unit cost for total usage [3], this difference can be significant.

This suggests a second use for the battery: to store *purchased* energy for future use. Indeed, at least one start-up company [1] is currently marketing such a battery-based system intended to reduce peak energy charges. In such a system, a battery is placed between the power company and a high-consumption client site, in order to smooth power requests and shave the peak. The client site will charge to the battery when demand is low and discharge when demand is high. Spikes in the demand curve can thus be rendered consistent with a relatively flat level of supplied power. The result is a lower cost for the client and a more manageable request curve for the provider.

We may generalize this problem of minimaxing the request to any resource which is *tenable* in the sense that it may be obtained early and stored until needed. For example, companies frequently face shortages of popular products: "Plentiful supply [of Xboxes] would be possible only if Microsoft made millions of consoles in advance and stored them without releasing them, or if it built vast production lines that only ran for a few weeks–both economically unwise strategies," a recent news story asserted [11]. A producer could smooth the product production curve by increasing production and warehousing supply until future sales. But when should the producer "charge" and "discharge"? (In some domains, there may also be an unpredictable level of volunteer help.) A third application is the scheduling of jobs composed of generic work-units that may be done in advance. Although the problem is very general, we will use the language of energy and batteries for concreteness.

In the online version of our problem, the essential choice faced at each timeslot is whether (and by how much) to invest in the future or to cash in a prior investment. The investment in our setting is a request for more energy than is needed at the time. If the algorithm only asks for the minimum required, then it is vulnerable to spikes in demand; if it asks for much more energy than it needs, then the greater request could itself introduce a new, higher peak. The strictness of the problem lies in the fact that the cost is not cumulative: we want *every* request to be low.

Background. Experimental work applying variations of these online algorithms to settings lacking provable guarantees was recently presented [6]. The present paper focuses on settings that allow guaranteed competitiveness.

There is a wide literature on commodity production, storage, warehousing, and supply-chain management (see e.g. [13, 17, 9, 14]). More specifically, there are a number of inventory problems based on the Economic Lot Sizing model [8], in which demand levels for a product vary over a discrete finite time-horizon and are known in advance. A feasible solution in these problems must obtain sufficient supply through production (sometimes construed as ordering) or through other methods, in order to meet each of the demands on time, while observing certain constraints. The nature of solution quality varies by formulation.

One such inventory problem is Single-Item Lot-Sizing, in which sufficient supplies must be ordered to satisfy each demand, while minimizing the total cost of ordering

charges and holding charges. The ordering charge consists of a fixed charge per order plus a charge linear in order size. The holding charge for inventory is per-unit and per-timeslot. There is a tradeoff between these incentives since fixed ordering charges encourage large orders while holding charges discourage them. Wagner & Whitin [15] showed in 1958 that this problem can be solved in polynomial time. Under the assumption of *non-speculative costs*, in which case orders should always be placed as late as possible, the problem can be solved in linear time. Such "speculative" behavior, however, is the very motivation of our problem. There are many lot-sizing variations, including constant-capacity models that limit the amount ordered per timeslot. (See [14] and references therein.) Our offline problem differs in that our objective is minimizing this constant capacity (for orders), subject to a bound on *inventory* size, and we have no inventory charge.

Another related inventory problem is Capacity and Subcontracting with Inventory (CSI) [5], which incorporates trade-offs between production costs, subcontracting costs, holding costs, and the cost for maximum per-unit-timeslot production capacity. The goal in that problem is to choose a production capacity and a feasible production/ subcontracting schedule that together minimize total cost, whereas in our problem choosing a production capacity, subject to storage constraints, is the essential task.

In the minimax work-scheduling problem [12], the goal is to minimize the maximum amount of work done in any timeslot over a finite time-horizon. Our online problem is related to a previously studied special case in which jobs with deadlines are assigned online. In that problem, all work must be done by deadline but cannot be begun until assigned. Subject to these restrictions, the goal is to minimize the maximum work done in any timeslot. While the optimization goal is the same, our online problem differs in two respects. First, each job for us is due immediately when assigned. Second, we *are* allowed to do work (request and store energy) in advance. One online algorithm for the jobs-by-deadlines problem is the α*-policy* [12]: at each timeslot, the amount of work done is α *times the maximum per-unit-timeslot amount of work that OPT would have done, when running on the partial input received so far*. One of our online algorithms adopts a similar strategy.

Contributions. We introduce a novel scheduling problem and solve several versions optimally with efficient combinatorial algorithms. We solve the offline problem for two kinds of batteries: unbounded battery in $O(n)$ time and bounded in $O(n^2)$. Separately, we show how to find the optimal offline battery size, for the setting in which the final battery level must equal the initial battery level. This is the smallest battery size that achieves the optimal peak. The online problem we study is very strict. A meta-strategy in many online problems is to balance expensive periods with cheap ones, so that the overall cost stays low [7]. The difficulty in our problem lies in its non-cumulative nature: we optimize for the max, not for the average. We show that several versions of the online problem have no algorithm with non-trivial competitive ratio (i.e., better than n or $\Omega(\sqrt{n})$). Given advanced knowledge of the peak demand D, however, we give H_n-competitive algorithms for batteries bounded and unbounded. Our fastest algorithm has $O(1)$ per-slot running-time. H_n is the (optimal) competitive ratio for both battery settings.

Examples. Although there is no constant-ratio competitive algorithm for unbounded n, our intended application in fact presumes a fixed time-horizon. If the billing period is one month, and peak charges are computed as 30-minute averages, then for this setting H_n is approximately 7.84. If we assume that the battery can fully recharge at night, so that each day can be treated as a separate time period, then for a 12-hour daytime time-horizon H_n is approximately 3.76.

2 Model and preliminaries

Definition 1. *At each timeslot i, d_i is the demand, r_i is the request, b_i is the battery charge level* at the start *of the timeslot, and f_i is the amount of free source available. By $\hat{d}_i$ we indicate the* effective demand $d_i - f_i$. *We sometimes refer to the sequence over time of one of these value types as a curve, e.g., the* demand curve. *D is the maximum effective demand* $\max_i\{\hat{d}_i\}$, *and R is the maximum request* $\max_i\{r_i\}$.

The problem instance comprises the demands, the free source curve, battery size B, initial charge b_1, and required final charge b_{n+1} (in the offline case). The problem solution consists of the request curve.

Definition 2. *Let* overflow *be the situation in which $r_i + f_i - d_i > B - b_i$, i.e., there is not enough room in the battery for the amount we want to charge. Let* underflow *be the situation in which $d_i - r_i - f_i > b_i$, i.e., there is not enough energy in the battery for the amount we want to discharge. Call an algorithm* feasible *if underflow never occurs. The goal of the problem is to minimize R (for competitiveness measures this is construed as maximizing $D - R$) while maintaining feasibility.*

In the absence of overflow/underflow, the battery level at timeslot i is simply $b_i = b_{i-1} + r_{i-1} + f_{i-1} - d_{i-1}$. It is forbidden for b_i to ever fall below 0. That is, the request r_i, the free source f_i, and the battery level b_i must sum to at least the demand d_i at each timeslot i. Notice that effective demand can be negative, which means that the battery may be charged (capacity allowing), even if the request is 0. We assume D, however, is strictly positive. Otherwise, the problem instance is essentially trivial. We use the following to simplify the problem statement:

Observation 1 *If effective demands may be negative, then free source energy need not be explicitly considered.*

As such, we set aside the notion of free source, and for the remainder of the paper (simplifying notation) allow demand d_i to be negative.

In the energy application, battery capacity is measured in kWh, while instantaneous request is measured in kW. By discretizing we assume wlog that battery level, demand, and request values are expressed in common units. Peak charges are based linearly on the max request, which is what we optimize for. The battery can have a *maximum capacity B* or be unbounded. The problem may be online, offline, or in between; we consider

the setting in which the peak demand D is revealed in advance, perhaps predicted from historical information.

Threshold algorithms: For a particular snapshot (d_i, r_i, b_i), demand d_i must be supplied through a combination of the request r_i and a change in battery $b_{i+1} - b_i$. This means that there are only three possible modes for each timeslot: request exactly the demand-free, request more than this and *charge* the difference, or request less and *discharge* the difference. We refer to our algorithms as *threshold algorithms*. Let $T_1, T_2, ..., T_n$ be a sequence of values. Then the following algorithm uses these as request thresholds:

```
for each timeslot i
    if d_i < T_i
        charge min(B − b_i, T_i − d_i)
    else
        discharge d_i − T_i
```

Intuitively, the algorithm amounts to the rule: *at each timeslot i, request an amount as near to T_i as the battery constraints will allow*. Our offline algorithms are *constant threshold* algorithms, with a fixed T; our online algorithms compute T_i dynamically for each timeslot i.

A constant-threshold algorithm is specifiable by a single number. In the online setting, predicting the *exact* optimal threshold from historical data suffices to solve the online algorithm optimally. A small overestimate of the threshold will merely raise the peak cost correspondingly higher. Unfortunately, however, examples can be found in which even a small *underestimate* eventually depletes the battery before peak demand and thus produce no cost-savings at all.

The *offline* problem can be solved approximately, within additive error ϵ, through binary search for the minimum feasible constant threshold value T. Simply search the range $[0, D]$ for the largest value T for which the threshold algorithm has no underflow, in time $O(n \log \frac{D}{\epsilon})$. If the optimal peak reduction is $R - T$, then the algorithm's peak reduction will be at least $R - T - \epsilon$. It is straightforward to give a linear programming formulation of the *offline* problem; it can also be solved by generalized *parametric* max-flow [4]. Our interest here, however, is in efficient combinatorial optimal algorithms. Indeed, our combinatorial offline algorithms are significantly faster than these general techniques and lead naturally to our competitive online algorithms. Online algorithms based on such general techniques would be intractable for fine-grain timeslots.

3 Offline problem

We now find optimal algorithms for both battery settings. For unbounded, we assume the battery starts empty; for bounded, we assume the battery starts with amount B. For both, the final battery level is unspecified. We show below that these assumptions are made wlog. The two offline threshold functions, shown in Table 1, use the following definition:

Definition 3. *Let $\mu(j) = \frac{1}{j} \sum_{t=1}^{j} d_t$ be the mean demand of the prefix region $[1, j]$,*

and let $\hat{\mu}(k) = max_{1 \le j \le k} \mu(j)$ *be the maximum mean among of the prefix regions up to* k*. Let* $\rho(i,j) = \frac{-B + \sum_{t=i}^{j} d_t}{j-i+1}$ *be the* density *of the region* $[i,j]$ *and* $\hat{\rho}(k) = max_{1 \le i \le j \le k} \rho(i,j)$ *be the maximum density among all subregions of* $[1,k]$.

Alg.	battery	threshold T_i	run-time
1.a	unbounded	$\hat{\mu}(n)$	$O(n)$
1.b	bounded	$\hat{\rho}(n)$	$O(n^2)$

Table 1. Threshold functions used for offline algorithm settings.

Bounded capacity changes the character of the offline problem. It suffices, however, to find the peak request made by the optimal algorithm, R_{opt}. Clearly $R_{opt} \ge D - B$, since the ideal case is that a width-one peak is reduced by size B. Of course, the peak region might be wider.

Theorem 2. *Algorithm* 1.a *(threshold* $T_i = \hat{\mu}(n)$*, for unbounded battery) and Algorithm* 1.b *(threshold* $T_i = \hat{\rho}(n)$*, for bounded battery) are optimal, feasible, and run in times* $O(n)$ *and* $O(n^2)$*, respectively.*

Proof. First, let the battery be unbounded. For any region $[1,j]$, the best we can hope for is that requests for all demands $d_1, ..., d_j$ can be spread evenly over the first j timeslots. Therefore the optimal threshold cannot be lower than the maximum $\mu(j)$, which is Algorithm *1.a*'s threshold. For feasibility, it suffices to show that after each time j, the battery level is nonnegative. But by time j, the total input to the system will be $j \cdot \hat{\mu}(n) \ge j \cdot \mu(j) = \sum_{t=1}^{j} d_j$, which is the total output to the system up to that point. For complexity, just note that $\mu(j+1) = \frac{j \cdot \mu(j) + d_{j+1}}{j+1}$, so the sequence of μ values, and their max, can be computed in linear time.

Now let the battery be bounded. Over the course of any region $[i,j]$, the best that can be hoped for is that the peak request will be reduced to $B/(j-i+1)$ less than the average d_i in the region, i.e., $\rho(i,j)$, so no threshold lower than Algorithm *1.b*'s is possible.

For feasibility, it suffices to show that the battery level will be nonnegative after each time j. Suppose j is the first time underflow occurs. Let $i-1$ be the last timeslot prior to j with a full battery. Then there is no underflow *or* overflow in $[i,j)$, and so for each $t \in [i,j]$ the discharge at t is $b_t - b_{t+1} = d_t - T$ (possibly negative, meaning a charge) and the so the total net discharge over $[i,j]$ is $\sum_{t=i}^{j} d_t - (j-i+1)T$. Total net discharge greater than B implies $T < \frac{-B + \sum_{t=i}^{j} d_t}{j-i+1}$, which contradicts the definition of T. The densest region can be found in $O(n^2)$, with n separate linear-time passes, each of which finds the densest region beginning in some position i, since $\rho(i,j+1)$ can be computed in constant time from $\rho(i,j)$.

3.1 Battery level boundary conditions

We assumed above that the battery starts empty for the unbounded offline algorithm and starts full for the bounded offline algorithm, with the final battery level left inde-

terminate for both settings. A more general offline problem may require that $b_1 = \beta_1$ and $b_{n+1} = \beta_2$, i.e., the battery begins and ends at some charge levels specified by parameters β_1 and β_2. We argue here that these requirements are not significant algorithmically, since by pre- and postprocessing, we can reduce to the default cases for both the unbounded and bounded versions.

First, consider the unbounded setting, in which the initial battery level is 0. In order to enforce that $b_1 = \beta_1$ and $b_{n+1} = \beta_2$, run the usual optimal algorithm on the sequence $(d_1 - \beta_1, d_2, ..., d_{n-1}, d_n + \beta_2)$. (Recall that negative demands are allowed.) Then b_{n+1} will be at least β_2 larger than d_n. To correct for any surplus, manually delete a total of $b_{n+1} - \beta_2$ from the final requests. For the bounded setting, the default case is $b_1 = B$ and b_{n+1} indeterminate. To support $b_1 = \beta_1 \neq B$ and $b_{n+1} = \beta_2$, modify the demand sequence as above, except with $d_1 + (B - \beta_1)$ as the first demand and then do similar postprocessing as in the unbounded case to deal with any final surplus.

3.2 Optimal battery size

A large component of the fixed initial cost of the system will depend on battery capacity. A related problem therefore is finding the optimal battery size B for a given demand curve d_i, given that the battery starts and ends at the same level β (which can be seen as an amount *borrowed and repaid*). The optimal peak request possible will be $\frac{1}{n}\sum_{i=1}^{n} d_i = \mu(n)$, and the goal is to find the smallest B and β that achieve peak $\mu(n)$. (A completely flat request curve is possible given a sufficiently large battery.) This can be done in $O(n)$.

Since we will have $b_1 = b_{n+1}$, $r_i = \mu$ for all i, and $\sum d_i = \sum r_i$, there must be no overflow. Let $d'_i = d_i - \mu$, i.e., the amount by which d_i is above average (positive means discharge, negative means charge). Then the minimum possible $\beta = b_1$ is the maximum prefix sum of the d' curve (which will be at least 0). It could happen that the battery level will at some point rise above b_1, however. (Consider the example $d = (0, 0, 1, 0, 0, 0)$, for which $\mu = 1/6$, $d' = (-1/6, -1/6, 5/6, -1/6, -1/6, -1/6)$ and $\beta = 1/2$.) The needed capacity B can be computed as β plus the maximum prefix sum of *the negation of* the d' curve (which will also be at least 0). (In the example, we have $B = \beta + 2/6 = 5/6$.)

Although B is computed using β, we emphasize that the computed β is the minimum possible *regardless of* B, and the computed B is the minimum possible *regardless of* β.

4 Online problem

We consider two natural choices of objective function for the online problem. One option is to compare the peak requests, so that if ALG is the peak request of the online algorithm ALG and OPT is that of the optimal offline algorithm OPT, then a c-competitive algorithm for $c \geq 1$ must satisfy $\frac{ALG}{OPT} \leq c$ for every demand sequence. Although this may be the most natural candidate, we argue that for many settings it is

uninteresting. If the peak demand is a factor k larger than the battery capacity, for example, then the trivial online algorithm that *does not use* the battery would be $(k/(k-1))$-competitive. If we drop the assumption of a small battery, then no reasonable competitive factor is possible in general, *even if D is revealed in advance*.

Proposition 1. *With peak demand D* revealed in advance *and finite time horizon n, no online algorithm* for the problem of minimizing peak request *can have competitive ratio 1) better than n if the battery begins with some strictly positive charge, or 2) better than $\Omega(\sqrt{n})$ if the battery begins empty.*[1]

Proof. For part 1, suppose that the battery begins with initial charge D. Consider the demand curve $(D, 0, ..., 0, ?)$, where the last demand is either D or 0. ALG *must* discharge D at time 1, since $OPT = 0$ when $d_n = 0$. Thus ALG's battery is empty at time 2. If ALG requests nothing between times 2 and $n-1$, and $d_n = D$, then we have $OPT = D/n$ and $ALG = D$; if ALG requests some $\alpha > 0$ during any of those timeslots, and $d_n = 0$, then we have $OPT = 0$ and $ALG = \alpha$. This yields a lower bound of n.

For part 2, suppose the battery begins empty, which is a disadvantage for both ALG and OPT. Consider the demand curve $(0, 0, ..., 0, D)$, in which case $OPT = D/n$. If an algorithm is c-competitive for some $c \geq 1$, then in each of the first $n-1$ timeslots of this demand curve ALG can charge at most amount cD/n. Now suppose that the only nonzero demand, of value D, arrives possibly earlier, at some timeslot $k \in [1, n]$, following $k-1$ demands of zero, during which ALG can charge at most $(k-1)cD/n$. In this case, we have $OPT = D/k$ and $ALG \geq D - (k-1)cD/n$, which yields the competitive ratio:

$$c \geq \frac{D - (k-1)cD/n}{D/k} = \frac{1 - (k-1)c/n}{1/k} = k - k^2c/n + kc/n$$

Solving for c, and then choosing $k = \sqrt{n}$, we have:

$$c \geq \frac{k}{1 + k^2/n - k/n} = \frac{\sqrt{n}}{1 + 1 - 1/\sqrt{n}} = \Omega(\sqrt{n})$$

thus establishing the lower bound.

Instead, we compare the *peak shaving amount* (or *savings*), i.e., $D - R$. For a given input, let *OPT* be the peak shaving of the optimal algorithm, and let *ALG* be the peak shaving of the online algorithm. Then an online algorithm is c-competitive for $c \geq 1$ if $c \geq \frac{OPT}{ALG}$ for every problem instance. For this setting, we obtain the online algorithms described below.

The online algorithms (see Def. 4) are shown in Table 2.

Definition 4. *Let T_i^{opt} be the optimal threshold used by the appropriate optimal algorithm when run on the first i timeslots. At time i during the online computation, let s_i be the index of the most recent time prior to i with $b_{s_i} = B$ (or 1 in the unbounded setting).*

[1] If the peak is not known, a lower bound of n can be obtained also for the latter case.

Alg.	battery	threshold T_i	per-slot time
2.a	both	$D - \frac{D-T_i^{opt}}{H_n}$	$O(n)$
2.b	both	$D - \frac{D-\rho(s_i,i)}{H_{n-s_i+1}}$	$O(1)$

Table 2. Threshold functions used for online algorithms.

4.1 Lower bounds for $D - R$

Since the competitiveness of the online algorithms holds for arbitrary initial battery level, in obtaining lower bounds on competitiveness, we assume particular initial battery levels.

Proposition 2. *With peak demand D unknown and finite time horizon n, there is no online algorithm 1) with any constant competitive ratio for unbounded battery (even with $n = 2$) or 2) with competitive ratio better than n for bounded battery.*

Proof. For part 1, assume $b_1 = 0$, and suppose $d_1 = 0$. Then if ALG requests $r_1 = 0$ and we have $d_2 = D$, then $OPT = D/2$ and $ALG = 0$; if ALG requests $r_1 = a$ (for some $a > 0$) and we have $d_2 = a$, then $OPT = a/2$ and $ALG = 0$. For part 2, let $b_1 = B$, and assume ALG is c-competitive. Consider the demand curve $(B, 0, 0, \ldots, 0)$. Then OPT clearly discharges B at time 1 (decreasing the peak by B). For ALG to be c-competitive, it must discharge at least $\frac{B}{c}$ in the first slot. Now consider curve $(B, 2B, 0, 0, \ldots, 0)$. At time 2, OPT discharges B, decreasing the peak by B, so at time 2, ALG must discharge at least $\frac{B}{c}$. (At time 1, ALG already had to discharge $\frac{B}{c}$.) Similarly, at time i for $(B, 2B, 3B, ..., iB, ..., nB)$, ALG must discharge $\frac{B}{c}$. Total discharging by ALG is then at least: $\sum_{i=1}^{n} \frac{B}{c} = \frac{nB}{c}$. Since we must have $n\frac{B}{c} \leq B$, it follows that $c \geq n$.

The trivial algorithm that simply discharges amount B/n at each of the n timeslots and *never charges* is n-competitive (since $OPT \leq B$) and so matches the lower bound for the bounded case.

Proposition 3. *With peak demand D known in advance and finite time horizon n, no online algorithm can have 1) competitive ratio better than H_n if the battery begins nonempty or 2) competitive ratio better than $H_n - 1/2$ if the battery begins empty, regardless of whether the battery is bounded or not.*

Proof. First assume the battery has initial charge b. (The capacity is either at least b or unbounded.) Suppose ALG is c-competitive. Consider the curve $(D, 0, 0, \ldots, 0)$, with $D \geq b$. Then OPT clearly discharges b at time 1 (decreasing the peak by b). For ALG to be c-competitive, it must discharge at least $\frac{b}{c}$. Now consider curve $(D, D, 0, 0, \ldots, 0)$. At times 1 and 2, OPT discharges $\frac{b}{2}$, decreasing the peak by $\frac{b}{2}$. At time 2, ALG will have to discharge at least $\frac{b/2}{c} = \frac{b}{2c}$. Similarly, at time i on $(D, D, D, ..., D)$, ALG must discharge $\frac{b}{ic}$. Total discharging by ALG is then at least: $\sum_{i=1}^{n} \frac{b}{ic} = H_n \frac{b}{c}$. Since we

discharge at each timeslot and never charge, we must have $\frac{b}{c}H_n \leq b$, and so it follows that $c \geq H_n$.

Now let the battery start empty. Assume the battery capacity is at least D or is unbounded. Repeat the argument as above, except now with a zero demand inserted at the start of the demand curves, which gives both ALG and OPT an opportunity to charge. Then for each time $i \in [2, n]$, ALG must discharge at least $\frac{D}{ic}$ since OPT may discharge (and so save) $\frac{D}{i}$ (in which case it would have initially charged $D(1 - 1/i)$). ALG is then required to discharge $(H_n - 1)\frac{D}{c}$ during the last $n - 1$ timeslots. Obviously it could not have charged more than D during the first timeslot. In fact, it must charge less than this. On the sequence $(0, D, 0, 0, ..., 0)$, OPT charges $D/2$ at time 1 and discharges it at time 2, saving $D/2$. ALG must discharge $\frac{D}{2c}$ at time 2 in order to be c-competitive on this sequence, and so reduce the peak D by $\frac{D}{2c}$. Therefore at time 1, ALG cannot charge more than $D - \frac{D}{2c}$. Therefore we must have $D - \frac{D}{2c} \geq (H_n - 1)D/c$, which implies that $c \geq H_n - 1/2$.

4.2 Bounded battery

Our first online algorithm bases its threshold at time i on a computation of the optimal offline threshold T_i^{opt} for the demands $d_1, ..., d_i$. The second bases its threshold at time i on $\rho(s_i, i)$ (see Defs. 3 and 4). Assuming the algorithms are feasible (i.e., no battery underflow occurs), it is not difficult to show that they are competitive.

Theorem 3. *Algorithms* 2.a *and* 2.b *are H_n-competitive, if they are feasible, and have* per-timeslot *running times of $O(n)$ and $O(1)$, respectively.*

Proof. First observe that $\hat{\rho}(i) \geq \rho(s_i, i)$ implies $\frac{D - \hat{\rho}(i)}{H_n} \leq \frac{D - \rho(s_i, i)}{H_{n - s_i + 1}}$ implies $T_i^a \geq T_i^b$ for all i. Therefore it suffices to prove competitiveness for Algorithm *2.a*. Since T_i^{opt} is the lowest possible threshold up to time i, $D - T_i^{opt}$ is the highest possible peak shaving as of time i. Since the algorithm always saves a $1/H_n$ fraction of this, it is H_n-competitive by construction.

Since $\mu(1, i + 1)$ can be found in constant time from $\mu(1, i)$, Algorithm *2.b* is constant-time per-slot. Similarly, Algorithm *2.a* is, recalling the proof of Theorem 2, linear per-slot.

We now show that indeed both algorithms are feasible, using the following lemma, which allows us to limit our attention to a certain family of demand sequences.

Lemma 1. *If there is a demand sequence $(d_1, d_2, ..., d_n)$ in which underflow occurs for Algorithm* 2.a *or* 2.b*, then there is also a demand sequence (for the same algorithm) in which underflow continues to the end (i.e., $b_{n+1} < 0$) and no overflow ever occurs, i.e., one in which the battery level decreases monotonically from full to empty.*

Proof. The battery is initialized to full, $b_1 = B$. Over the course of running one of the algorithms on a particular problem instance, the battery level will fall and rise, and may

return to full charge multiple times. Suppose underflow were to occur at some time t, i.e. $b_t < 0$, and let s be the most recent time before t when the battery was full. We now construct a demand sequence with the two desired properties, for both algorithms.

First, if $s > 1$, then also considering region $[1, s-1]$ when defining the threshold T_i for Algorithm *2.a* or *2.b* can only raise the threshold over what it would be if only region $[s, t]$ were considered. Therefore shifting the region leftward from $[s, t]$ to $[1, t'] = [1, t - s + 1]$ will only lower the thresholds used, which therefore preserves the underflow. Second, since any underflow that occurs in region $[1, t']$ can be extended to the end of sequence by setting each demand after time t' to D, we can assume wlog that $t' = n$.

Theorem 4. *Algorithms* 2.a *and* 2.b *are feasible.*

Proof. For a proof by contradiction, we can restrict ourselves by Lemma 1 to regions that begin with a full battery, underflow at the end, and have no overflow in the middle. For such a region, the change in battery-level is well behaved ($b_i - b_{i+1} = d_i - T_i$), which allows us to sum the net discharge and prove it is bounded by B. We now show that it is impossible for the battery to fall below 0 at time n, by upperbounding the *net discharge* over this region. Let $\Delta b_i = b_i - b_{i+1} = d_i - T_i$ be the amount of energy the battery discharges at step i. (Δb_i will be negative when the battery charges.) We will show that $\sum_{1 \le i \le n} \Delta b_i \le B$. Let Δb_i^a and Δb_i^b refer to the change in battery levels for the corresponding algorithms. Because as we observed above $T_i^a \ge T_i^b$, we have:

$$\Delta b_i^a = d_i - T_i^a \le \Delta b_i^b = d_i - T_i^b$$

Therefore it suffices to prove the feasibility result for Algorithm *2.b*, and so we drop the superscripts. Expanding the definition of that algorithm's threshold, we have:

$$\Delta b_i = d_i - T_i = d_i - \left(D - \frac{1}{H_n}(D - \rho(1,i))\right) = d_i - \left(D - \frac{1}{H_n}\left(D - \frac{1}{i}\left(\sum_{k=1}^{i} d_k - B\right)\right)\right) \tag{1}$$

By summing Eq. 1 for each i, we obtain:

$$\begin{aligned}
\sum_{i=1}^{n} \Delta b_i &= \sum_{i=1}^{n} \left(d_i - \left(D - \frac{D - (\sum_{k=1}^{i} d_k - B)/i}{H_n}\right)\right) \\
&= \sum_{i=1}^{n} \left(d_i - \left(D - \frac{D - (\sum_{k=1}^{i} d_k)/i}{H_n}\right)\right) + \sum_{i=1}^{n} \frac{B/i}{H_n} \\
&= \sum_{i=1}^{n} \left(d_i - \left(D - \frac{D - (\sum_{k=1}^{i} d_k)/i}{H_n}\right)\right) + B
\end{aligned}$$

Therefore it suffices to show that:

$$\sum_{i=1}^{n} \left(d_i - \left(D - \frac{D - \sum_{k=1}^{i} d_k/i}{H_n}\right)\right) \le 0 \tag{2}$$

which is equivalent to:

$$\sum_{i=1}^{n} d_i - nD + \frac{1}{H_n}\Big(nD - \sum_{i=1}^{n}\sum_{k=1}^{i} d_k/i\Big) \leq 0$$

$$\textit{iff } \sum_{i=1}^{n} d_i + nD(\frac{1}{H_n} - 1) - \frac{1}{H_n}\Big(\sum_{i=1}^{n}\sum_{k=1}^{i} d_k/i\Big) \leq 0$$

$$\textit{iff } \sum_{i=1}^{n} H_n d_i - \Big(\sum_{i=1}^{n}\sum_{k=1}^{i} \frac{d_k}{i}\Big) \leq nD(H_n - 1) \tag{3}$$

With the following derivation:

$$\sum_{i=1}^{n}\sum_{k=1}^{i} \frac{d_k}{i} = \sum_{i=1}^{n}\sum_{k=1}^{n} \frac{d_k}{i} - \sum_{i=1}^{n}\sum_{k=i+1}^{n} \frac{d_k}{i} = \sum_{k=1}^{n}\sum_{i=1}^{n} \frac{d_k}{i} - \sum_{k=1}^{n}\sum_{i=1}^{k-1} \frac{d_k}{i} = \sum_{k=1}^{n} H_n d_k - \sum_{k=1}^{n} H_{k-1} d_k$$

we can rewrite Eq. 3 (replacing the parenthesized expression) as: $\sum_{i=1}^{n} H_{i-1} d_i \leq n(H_n - 1)D$. Since $d_i \leq D$ and $D > 0$, it suffices to show that: $\sum_{i=1}^{n} H_{i-1} \leq n(H_n - 1)$. In fact, this holds with equality (see [10], Eq. 2.36).

4.3 Unbounded battery and boundary conditions

Both online algorithms, modified to call appropriate subroutines, also work for the unbounded battery setting. The algorithms are feasible in this setting since $\rho(i) \leq T_i^{opt}$ still holds, where T_i^{opt} is now the optimal threshold for the unbounded battery setting. (Recall that offline Algorithm *1.a* can "greedily" run in linear total time.) The algorithm is H_n-competitive by construction, as before.

Corollary 1. *Algorithms* 2.a *and* 2.b *are feasible in the unbounded battery setting.*

Proof. The proof is similar to that of Theorem 4, except that b_1 (which may be 0) is plugged in for all occurrences of B (resulting in a modified ρ), and overflow is no longer a concern.

We also note that the correctness and competitiveness of both algorithms (with minor preprocessing) holds for the setting of arbitrary initial battery levels. In the case of Algorithm *2.a*, each computation of T_i^{opt} is computed for the chosen b_1, as described in Section 3.1, and the online algorithm again provides a fraction $1/H_n$ of this savings. Although in the bounded setting "increasing" b_1 for the T_i^{opt} computation by modifying d_1 may raise the peak demand *in the offline algorithm's input*, the D value for the online algorithm is not changed. Indeed, examining the proof of Theorem 4, we note that the upperbound on the d_i is only used for $i > 1$ (see Eq. 4).

5 Conclusion

In this paper, we formulated a novel peak-shaving problem, and gave efficient optimal offline algorithms and optimally competitive online algorithms. In work in progress, we are testing our online algorithms on actual client data from Gaia [1]. (See [6] for preliminary results.) There are several interesting extensions to the theoretical problem that we plan to address, such as adapting the algorithms to inefficient batteries that lose a percentage of charge instantly or over time or batteries with a charging speed limit. We could also optimize for a moving average of demands rather than a single peak. Finally, online algorithms could also be granted additional predictions about the future.

Acknowledgements: We thank Ib Olson of Gaia Power Technologies and Ted Brown of CUNY and its Institute for Software Development and Design for posing this problem to us.

References

1. Gaia Power Technologies. gaiapowertech.com.
2. ConEd electricity rates document. www.coned.com/documents/elec/043-059h.pdf.
3. Orlando Utilities Commission website. www.ouc.com/account/rates/electric-comm.htm.
4. R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows*. Prentice Hall, 1993.
5. A. Atamturk and D. S. Hochbaum. Capacity acquisition, subcontracting, and lot sizing. *Management Science*, Vol. 47, No. 8, 2001.
6. A. Bar-Noy, Y. Feng, M. P. Johnson, and O. Liu. When to reap and when to sow: Lowering peak usage with realistic batteries. In *WEA 2008*.
7. A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 1998.
8. M. Florian, J. Lenstra, and A. R. Kan. Deterministic production planning: algorithms and complexity. *Management Science*, Vol. 26, 1980.
9. M. Goh, O. Jihong, and T. Chung-Piaw. Warehouse sizing to minimize inventory and storage costs. *Naval Research Logistics*, Vol. 48, Issue 4, 3 Apr 2001.
10. R. L. Graham, D. E. Knuth, and O. Patashnik. *Concrete Mathematics: A Foundation for Computer Science, 2nd Edition*. Addison-Wesley Professional, 1994.
11. T. Harford. The great Xbox shortage of 2005. *Slate*, Dec. 15, 2005 www.slate.com/id/2132071/.
12. B. Hunsaker, A. J. Kleywegt, M. W. P. Savelsbergh, and C. A. Tovey. Optimal online algorithms for minimax resource scheduling. *SIAM J. Discrete Math.*, 2003.
13. M.-K. Lee and E. A. Elsayed. Optimization of warehouse storage capacity under a dedicated storage policy. *Int J Prod Res*, Vol. 43, No. 9, 2005.
14. Y. Pochet and L. Wolsey. *Production Planning by Mixed Integer Programming*. Springer, 2006.
15. H. Wagner and T. Whitin. Dynamic version of the economic lot size model. *Management Science*, Vol. 5, 1958.
16. M. L. Wald. Storing sunshine. *The New York Times*, July 16, 2007 www.nytimes.com/2007/07/16/business/16storage.html.
17. Y.-W. Zhou. A multi-warehouse inventory model for items with time-varying demand and shortages. *Computers and Operations Research*, Vol. 30, Issue 14, December 2003.