Demand Smoothing Through Resource Buffering

A. Bar-Noy  M.P. Johnson  O. Liu

City University of New York

Supported by NSF PFI and NYSTAR TIPP, and Gaia Power Technologies

October 4, 2008
Outline

1. Motivation
2. Offline problem
3. Online problem
4. Conclusion
Outline

1 Motivation

2 Offline problem

3 Online problem

4 Conclusion
High energy demand

- Key challenge for providers in electricity markets:
High energy demand

- Key challenge for providers in electricity markets:
  - High simultaneous demand
  - Limited supply (per unit time)
High energy demand

- Key challenge for providers in electricity markets:
  - High simultaneous demand
  - Limited supply (per unit time)

- ConEd wants to sell you lots of energy...
High energy demand

- Key challenge for providers in electricity markets:
  - High simultaneous demand
  - Limited supply (per unit time)

- ConEd wants to sell you lots of energy... but not all right now
Motivation

Offline problem

Online problem

Conclusion

High energy demand

Key challenge for providers in electricity markets:
- High simultaneous demand
- Limited supply (per unit time)

- ConEd wants to sell you lots of energy... but not all right now
- Extreme simultaneous usage is a challenge for the provider
- Difficult to prepare for, puts strain on grid, causes blackouts...
Demand charges

- One response by utilities: disincentivize peak usage
  - peak for the individual client
  - (other models: client incentives based on total current usage)
  - though this could be reposed from the provider’s pt of view
Demand charges

- One response by utilities: disincentivize peak usage
  - peak for the individual client
  - (other models: client incentives based on total current usage)
  - though this could be reposed from the provider’s pt of view

- Some large clients’ energy bills are based on both:
  - How much kWh electricity usage charges
  - How fast (at peak) kWh/h = kW power peak charges
Demand charges

- One response by utilities: disincetivize peak usage
  - peak for the individual client
  - (other models: client incentives based on total current usage)
  - though this could be reposed from the provider's pt of view

- Some large clients' energy bills are based on both:
  - How much kWh electricity usage charges
  - How fast (at peak) kWh/h = kW power peak charges

- Per-kW peak charge $\approx 100x$ per-kWh usage charge
  - Incentive: spread out usage over time
  - But not always possible - stores have customer surges, etc.
Alternative energy sources

- Energy sources such as solar, wind, etc., may be low-cost and clean...
Alternative energy sources

- Energy sources such as solar, wind, etc., may be low-cost and clean...
- but typically unpredictable.
- How to rely on them?
Solution to both: batteries

Figure: Gaia PowerTower
Other interpretations

- Resource is water
  - Power Tower → water tower

"The Xbox 360 can be produced only gradually, but all the demand is there at once. Plentiful supply 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. ... The steady supply can't match peak December demand." (http://www.slate.com/id/2132071/)

Or can it?

A. Bar-Noy, M.P. Johnson, O. Liu

Demand Smoothing Through Resource Buffering
Other interpretations

- Resource is water
  - Power Tower → water tower
- Resource is *undistinguished* unit-time jobs doable in advance
Other interpretations

- Resource is water
  - Power Tower → water tower
- Resource is *undistinguished* unit-time jobs doable in advance
- Resource is unsold products

The Xbox shortage of 2005

“[T]he Xbox 360 can be produced only gradually, but all the demand is there at once. Plentiful supply 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. ... The steady supply can’t match peak December demand.” (http://www.slate.com/id/2132071/)
Other interpretations

- Resource is water
  - Power Tower → water tower
- Resource is *undistinguished* unit-time jobs doable in advance
- Resource is unsold products

The Xbox shortage of 2005

“[T]he Xbox 360 can be produced only gradually, but all the demand is there at once. Plentiful supply 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. ... The steady supply can’t match peak December demand.” (http://www.slate.com/id/2132071/)

- *Or can it?*
Outline

1 Motivation

2 Offline problem

3 Online problem

4 Conclusion
Model and notation

- At each time: How much extra to request? Or how much less?

- Goal: make request curve as smooth as possible (\(\min \max\))
  - While always satisfying demand
  - Ideally without wasting any energy

- A dilemma:
  - Request nothing extra: waste the battery
  - Request too much extra: introduce new peaks

- Obj ftn is \(\max\), not sum
  - “strict liability”
Offline problem definition

Notation

- $n$: discrete timesteps
- $d_i$: demand at time $i$ (demands = input)
- $r_i$: request at time $i$ (requests = output)
- $D = \max_i d_i$
- $R = \max_i r_i$
- $b_i$: battery level at start of time $i$ ($b_1 = 0$ or $b_1 = B$)

Goal: choose requests $r_i$ to minimize $R$, i.e., make request curve as smooth as possible. All demands must be satisfied with no underflow: $\forall i \ b_i \geq 0$

NB: $b_{i+1} = b_i + r_i - d_i$ (except when underflow/overflow)
### Offline problem definition

#### Notation
- \( n \) discrete timesteps
- \( d_i \): demand at time \( i \) (demands = input)
- \( r_i \): request at time \( i \) (requests = output)
- \( D = \max_i d_i \)
- \( R = \max_i r_i \)
- \( b_i \): battery level at start of time \( i \) (\( b_1 = 0 \) or \( b_1 = B \))

#### Goal: choose requests \( r_i \) to minimize \( R \)
- i.e., make request curve as smooth as possible
  - all demands must be satisfied
  - with no underflow: \( \forall i \ b_i \geq 0 \)
- NB: \( b_{i+1} = b_i + r_i - d_i \) (except when underflow/overflow)
Incorporating the free source

- At each time may also have free source amount $f_i$
- In this case, effective demand is $\hat{d}_i = d_i - f_i$
- As long as negative demands make sense, can ignore free source wlog
Optimal solution (offline, unbounded battery)

Figure: Demands and mean
Optimal solution (offline, unbounded battery)

Figure: Demands, mean, and optimal
Threshold algorithms

- All our algorithms are based on *thresholds*
  - Threshold = amount the algorithm tries to request
  - Offline: global threshold $T$
  - Online: threshold $T_i$ at timestep $i$

- At each time, (try to) request $T_i$, and charge/discharge the rest (based on $d_i$ & $b_i$)
Threshold algorithms

- All our algorithms are based on *thresholds*
  - Threshold = amount the algorithm tries to request
  - Offline: global threshold $T$
  - Online: threshold $T_i$ at timestep $i$

- At each time, (try to) request $T_i$, and charge/discharge the rest (based on $d_i$ & $b_i$)

Two issues:

- **Overflow**: battery too full: ok, just lose the energy
  - Or just request less
- **Underflow**: battery below empty: forbidden ("crash")
Threshold algorithms

for each timeslot $i$
    if $T_i > d_i$
        charge $\min(T_i - d_i, B - b_i)$
    else
        discharge $d_i - T_i$

Figure: Threshold algorithm schema (assumes $b_i \geq d_i - T_i$)
Offline problems

- Two subsettings: unbounded and bounded batteries

- Both solvable by LP
  - But we seek efficient combinatorial algorithms
  - Our online algorithms will use offline as subroutine

- Initial/final conditions: slightly preprocess input (demands)
Offline problems

- Two subsettings: unbounded and bounded batteries

- Both solvable by LP
  - But we seek efficient combinatorial algorithms
  - Our online algorithms will use offline as subroutine
  - Initial/final conditions: slightly preprocess input (demands)

- Unbounded battery: find hardest prefix (average) of demands
  - For $b_1 = 0$ case
  - Easy in linear time
Offline problems

- Bounded battery: find hardest subsequence (critical region)
Offline problems

- Bounded battery: find hardest subsequence (*critical region*)
- In this region, for OPT:
  - battery will go from full to empty (if ever does)
  - requests will be flat
  - request value: \((-B + \sum_{t=i}^{j} d_t)/(j - i + 1)\) ("generalized average" or GA)
Offline problems

- Bounded battery: find hardest subsequence (critical region)
- In this region, for OPT:
  - battery will go from full to empty (if ever does)
  - requests will be flat
  - request value: \((-B + \sum_{t=i}^{j} d_t)/(j - i + 1)\) ("generalized average" or GA)

![Graph showing demand and generalized average]

A. Bar-Noy, M.P. Johnson, O. Liu

Demand Smoothing Through Resource Buffering
Offline problems

- Bounded battery: find hardest subsequence (critical region)
- In this region, for OPT:
  - battery will go from full to empty (if ever does)
  - requests will be flat
  - request value: \((-B + \sum_{t=i}^{j} d_t) / (j - i + 1)\) ("generalized average" or GA)
- Can easily find this region in quadratic time
Outline

1. Motivation
2. Offline problem
3. Online problem
4. Conclusion
Online algorithms

- Change: demands $d_i$ now arrive online (free source values, too)
- Goal: competitiveness with OPT
  - Potential obj ftns: minimize peak, or maximize savings
Online algorithms

- Change: demands $d_i$ now arrive online (free source values, too)
- Goal: competitiveness with OPT
  - Potential obj ftns: minimize peak, or maximize savings
- One idea: *alpha policy* [Hunsaker et al. 1998]
  - Common intuition: maybe the future will be like the past
Online algorithms

- Change: demands $d_i$ now arrive online (free source values, too)
- Goal: competitiveness with OPT
  - Potential obj ftns: minimize peak, or maximize savings
- One idea: $alpha$ policy [Hunsaker et al. 1998]
  - Common intuition: maybe the future will be like the past
    → at each moment, run OPT on the full history up until now
    - Then choose accordingly
      - i.e., request OPT’s max-so-far (times some $\alpha \geq 1$)
      - unbounded case: just the maximum prefix mean
Figure: Demands, mean, and optimal
Request graph: means

Figure: Demands, mean, and optimal
Request graph: maximean

Figure: Demands, mean, and optimal
Competitive online algorithm?

Unfortunately, there are competitiveness counterexamples for both the \textit{minimize peak} and \textit{maximize savings} problems.
Competitive online algorithm?

Unfortunately, there are competitiveness counterexamples for both the minimize peak and maximize savings problems.

Figure: Competitiveness counterexample for $b_1 = 0$ case
Semi-online algorithms

- So, relaxation for maximize savings problem: assume we can guess peak demand $D$ (e.g. from history data)
  - for minimize peak problem: still factor-$n$ lower bound on competitiveness
Semi-online algorithms

- So, *relaxation* for *maximize savings* problem: assume we can guess peak demand $D$ (e.g. from history data)
  - for *minimize peak* problem: still factor-$n$ lower bound on competitiveness

- Now do “alpha from above” (Alg 2.a)
- Since opt savings is $D - R_{opt}$
  - Always request so that savings is exactly $1/H_n$ of “optimal savings so far” (compared to $D$)
  - Alg 2.a: $T_i \leftarrow D - \frac{D - \hat{\mu}(1,i)}{H_n}$
Semi-online algorithms

- **So, relaxation** for *maximize savings* problem: assume we can guess peak demand $D$ (e.g. from history data)
  - for *minimize peak* problem: still factor-$n$ lower bound on competitiveness

- Now do “alpha from above” (Alg 2.a)

- Since opt savings is $D - R_{opt}$
  - Always request so that savings is exactly $1/H_n$ of “optimal savings so far” (compared to $D$)
  - Alg 2.a: $T_i \leftarrow D - \frac{D - \hat{\mu}(1,i)}{H_n}$

- $H_n$-competitive by construction, *assuming it’s correct*
  - i.e., assuming battery never crashes
  - i.e., request $T_i$ always suffices, with no underflow
Semi-online algorithms

Lemma

If there is an instance with underflow for Alg 2.a, then there will be one with battery decreasing from full to empty, with no overflow in the middle.
Semi-online algorithms

Lemma

If there is an instance with underflow for Alg 2.a, then there will be one with battery decreasing from full to empty, with no overflow in the middle.

In this case, we have: $b_{i+1} = b_i + r_i - d_i$. 
Semi-online algorithms

Lemma

If there is an instance with underflow for Alg 2.a, then there will be one with battery decreasing from full to empty, with no overflow in the middle.

In this case, we have: \( b_{i+1} = b_i + r_i - d_i \).

Thm: Alg 2.a is correct, and so \( H_n \)-competitive

Proof sketch: We have \( H_n \)-approx by construction, as long as no underflow. Cite lemma. But for such monotonic instances, total net discharge is \( \leq B \). Thus the final battery level is nonnegative.
Semi-online algorithms

**Lemma**
If there is an instance with underflow for Alg 2.a, then there will be one with battery decreasing from full to empty, with no overflow in the middle.

In this case, we have: \( b_{i+1} = b_i + r_i - d_i \).

**Thm:** Alg 2.a is correct, and so \( H_n \)-competitive

Proof sketch: We have \( H_n \)-approx by construction, as long as no underflow. Cite lemma. But for such monotonic instances, total net discharge is \( \leq B \). Thus the final battery level is nonnegative.

NB: applies to both bounded and unbounded battery.
Semi-online algorithms

Thm: $H_n$-competitive is optimal for bounded battery

Proof sketch: If ALG is $c$-competitive ($c \geq 1$) and $b_1 = B$, can force it to discharge:
- $B/c$ at time 1,
- $B/(2c)$ at time 2,
- $B/(3c)$ at time 3, etc.

Total discharge: $\sum_i B/(i \cdot c) = H_n \cdot B/c$.

The demand sequence is just: $(D, D, D, ..., D)$, for some $D \geq B$. 
Semi-online algorithms

**Thm:** $H_n$-competitive is optimal for bounded battery

Proof sketch: If ALG is $c$-competitive ($c \geq 1$) and $b_1 = B$, can force it to discharge:

- $B/c$ at time 1,
- $B/(2c)$ at time 2,
- $B/(3c)$ at time 3, etc.

Total discharge: $\sum_i B/(i \cdot c) = H_n \cdot B/c$.

The demand sequence is just: $(D, D, D, ..., D)$, for some $D \geq B$.

For unbounded battery, have lower bound of $H_n - 1/2$. 
Semi-online algorithms

- NB: *in some sense*, $n$ (hence $H_n$) is constant
  - monthly billing periods, coarseness of time units...
Semi-online algorithms

- NB: *in some sense*, $n$ (hence $H_n$) is constant
  - monthly billing periods, coarseness of time units...

- Alg 2.a can be $O(n^2)$
  - Basically computing OPT over time
  - At each time, extend $O(n)$ GAs
Semi-online algorithms

- NB: *in some sense*, $n$ (hence $H_n$) is constant
  - monthly billing periods, coarseness of time units...

- Alg 2.a can be $O(n^2)$
  - Basically computing OPT over time
  - At each time, extend $O(n)$ GAs

- Turns out (lemma) we can do better with less work: $O(n)$
  - Suffices to find the GA back to last time battery was full (for us)
  - Forget about prefix: $n \rightarrow n' < n$
  - More importantly: only one GA to extend each time
  - Alg 2.b: $T_i \leftarrow D - \frac{D - \mu(s_i,i)}{H(n-s_i+1)}$

- Analysis same as for Alg 2.a
Outline

1. Motivation
2. Offline problem
3. Online problem
4. Conclusion

A. Bar-Noy, M.P. Johnson, O. Liu
Demand Smoothing Through Resource Buffering
Summary

- Gave poly-time offline algorithms for bounded and unbounded batteries
- Unfortunately, many online problem settings here are intractable, but not all
- Gave $O(1)$-per-unit-time online algorithms for two of them
Future directions

Problem extensions:

- Entry loss
  - Corresponding online algorithm also appears to be $H_n$-competitive (WEA '08), but no proof
- Self-discharge (batteries draining over time)
- 30-minute rolling averages

Experimental work:

- Tuning more aggressive algorithms to empirical data (WEA '08)

Other models:

- E.g. dynamic pricing based on total current demand
Thanks!

mpjohnson@gmail.com