Predicting Peak-Demand Days in the Ontario Peak Reduction Program for Large Consumers

Yuheng Helen Jiang, Ryan Levman, Lukasz Golab and Jatin Nathwani
University of Waterloo, Canada
200 University Avenue West
Waterloo, Ontario, Canada
{y29jiang,rslevman,lgolab,nathwani}@uwaterloo.ca

ABSTRACT

In this paper, we propose a heuristic algorithm for day-ahead prediction of the top K days having the highest peak hourly demand for electricity over a given year. This problem, which arises in the context of critical peak pricing in Ontario, Canada, is difficult because we may have to wait till the end of the year to find out which K days ended up being the peak days. Our solution is to leverage short-term load forecasts and call tomorrow a peak day if it has sufficiently high probability of being a peak day in the time window covered by the forecast. Using Ontario demand data from 2005 till 2013, we show that our algorithm is more consistent from year to year and outperforms existing solutions to problems related to ours.

1. INTRODUCTION

Reducing peak electricity consumption is an important problem, which has led to a variety of peak pricing schemes in many jurisdictions. In this paper, we analyze the Five-Coincident-Peaks (5CP) program that affects large industrial and commercial consumers (whose monthly peak exceeds 5 megawatts) in the province of Ontario, Canada. In this program, large consumers pay heavy surcharges for the electricity they consumed during the five days with the highest peak hourly demand [1]. For some customers, these surcharges are higher than their volumetric charges [9].

The 5CP program is different to, e.g., California's Critical Peak Pricing (CPP) [3], in which utilities choose which days will be peak-pricing days according to some criteria, and they notify the participating customers in advance. In 5CP, Ontario's Electricity System Operator (IESO) waits till the end of the year and applies the surcharge to each large consumer based on its contribution to the load (at the peak hour) on the actual five peak days of the year. Without the benefit of hindsight, it is difficult for consumers to know when to curtail load in order to avoid surcharges.

We propose an algorithm that, at the end of every day, predicts whether tomorrow will be one of the five peak days of the current year, given only the publicly-available information such as short-term and long-term load forecasts and historical load statistics. We define the *precision* of such an algorithm as the fraction of days identified by it that are in fact peak days, and *recall* as the fraction of actual peak days that were identified as such by the algorithm. For example, suppose the actual peak days for some year were June 1, July 2, July 3, July 25 and August 8. Suppose that during the course of this year, the algorithm predicts the following *six* days as being peak days: June 1, July 1, July 2, July 3, July 20 and August 8. Its precision is $\frac{4}{6}$ and recall is $\frac{4}{5}$.

Obtaining perfect recall is easy: we predict that each day will be a peak day. Of course, precision will be very low and there will be many false alarms, causing customers to curtail operations unnecessarily and lose business. Ideally, we should achieve high precision (few false alarms) and high recall (few missed alarms).

The IESO publishes 12-month load forecasts, but they are not accurate because Ontario's peak demand is correlated with daily high temperatures, especially in the summer when the daily peak is caused by high air-conditioning use in the afternoon. The 14-day short-term forecasts are quite accurate, and the proposed algorithm uses these. For comparison, we also implemented three existing algorithms that solve related problems and may be adapted to our problem, including a technique based on historical load data, an algorithm based on stopping theory, and a dynamic programming algorithm that utilizes the full long-term forecast. We tested all four algorithms on Ontario's hourly demand data from 2005 to 2013, and found that our algorithm performs more consistently from year to year and obtains better recall while achieving similar precision.

The remainder of this paper is organized as follows. Section 2 discusses related work; Section 3 describes the proposed algorithm and extensions of three existing algorithms; Section 4 describes the experimental results; and Section 5 concludes the paper with directions for future work.

2. RELATED WORK

Ad-hoc solutions are often employed in the context of peak reduction programs such as 5CP and CPP. For instance, the University of Western Ontario, which is classified as a large customer, reduces its air conditioner usage during 2-6 p.m. from June 20 to August 31 in response to the 5CP program [5]. Other large consumers may try to reduce demand whenever tomorrow's peak demand forecast exceeds some threshold, e.g., 23,000 megawatts. Both of these methods sacrifice precision for recall.

An approach that uses historical peak demand data is used by a utility in California to decide which days to call in the CPP program [6]. Based on past data, this algorithm computes, on average, how many peak days tend to occur in each half-month. It then uses this distribution, along with temperature forecasts, to decide which days to call in the current year. We compare our algorithm against a variant of this technique in Section 3.

Predicting peak days in a day-ahead manner is related to stopping problems. For example, in the Secretary Selection problem, we interview up to n job applicants in random order, and, at the end of each interview, we decide (irrevocably) whether to hire this candidate or continue the interviews. The goal is to hire one best candidate. There exist extensions to multiple candidates (see, e.g., [7, 8, 10]), but they require knowledge of the probability distribution over the candidates. One exception is [2], which we extend to our problem and compare to our algorithm in Section 3.

There are also optimization algorithms for maximizing various functions in the context of California's CPP program, including scheduling CPP days by entities that sell wind energy [12] and maximizing savings functions [4, 11]. We implemented a variant of the most recent of these algorithms [4] and compare it to our algorithm in Section 3.

3. ALGORITHMS

We start by presenting three algorithms based on existing solutions to related problems: an algorithm based on that used by a California utility for the CPP program [6], an algorithm based on stopping theory [2], and an optimization algorithm that maximizes the sum of the forecasted peak demand values of the five predicted peak days [4]. We then discuss our algorithm.

At the beginning of the year, each algorithm is given the actual peak hourly demand for each day in the past year and the IESO 12-month long-term forecast for the current year. At the end of each day in the current year, we will be given the actual peak demand for that day, the 14-day short-term forecast from the IESO and the weather forecast for tomorrow. At the end of each day, we need to decide whether or not to classify tomorrow as a peak day[1]. Let D_i be the actual peak hourly demand on day i and let $\hat{D}_{iL}$ be the estimated peak hourly demand on day $i + L$ as of day i. Let K be the number of peak days we need to call, i.e., for 5CP, $K = 5$.

Each algorithm maintains a demand threshold τ_D, which is a lower bound for a peak day, i.e., any day whose peak demand forecast is below τ_D will not be called a peak day. Some algorithms will require an initial value of τ_D, which we set to be the peak demand of the Kth highest day in the previous year minus a small number that will be defined shortly.

We apply the following optimizations to each algorithm based on domain knowledge. First, only non-holiday weekdays can be peak days since the demand on weekends and holidays is much lower. Second, since Ontario has been a summer-peaking province since 2005, rather than running the algorithms for a whole year, we only run them from May 1 till September 30. Let N be the number of days in this interval.

3.1 The CPP Approach

The first algorithm which we call "CPP" analyzes historical peak demand data and computes, on average, how many of the K peak-demand days occur in each half-month. This distribution is then used in the current year as follows. If tomorrow's forecast exceeds the threshold τ_D, we predict that tomorrow will be a peak day. Now suppose that, based on historical data, there are on average two peak days by July 15. If we have already predicted two peak days, say, by July 1, then we raise τ_D by a small amount δ. If we did not yet predict any peak days and it is already July 14, then we lower τ_D by δ. Thus, the historical distribution of peak days throughout the year is used to dynamically adjust the threshold.

[1]In 5CP, large consumers only need to reduce load during the peak hour of one of the five peak days. We focus on predicting whether tomorrow will be a peak day, and predicting the peak hour can be done separately from the IESO day-ahead load forecast.

```
1. P = 0 // how many peak days called up to now
2. FOR i= 0 to N-1
3.   IF D̂_i1 > τ_D
4.     Predict "tomorrow will be a peak day"
5.     P = P + 1
6.   IF i is the 1st or 15th day of the month
7.     IF P > H[i]
8.       τ_D = τ_D + δ // raise threshold
9.     ELSE IF P < H[i]
10.      τ_D = τ_D − δ // lower threshold
```

Figure 1: The CPP Algorithm.

The pseudocode of the CPP algorithm is shown in Figure 1. The algorithm requires three additional inputs: an initial value of τ_D, a value for the threshold adjustment δ, and an array $H[i]$ with the average number of actual peak days that have occurred by day i in past years. As for δ, we can, for instance, use the average difference between the ith and the $i + 1$st ranked peak days from the previous year, for i between 1 and K. We will experiment with different values of τ_D and δ in Section 4.

The algorithm works as follows. Every day, we compare tomorrow's peak forecast to the current threshold (line 3). Additionally, on the 1st and the 15th of every month, we adjust the threshold by comparing the number of peak days we have called up to now with the average number of actual peak days that have occurred up to this day in previous years (lines 6-10).

3.2 The Stopping Approach

In stopping problems, we are given a sequence of numbers, one-by-one, and we need to decide when to stop examining the numbers and declare the current number to be the largest[2]. One solution is to view the first m numbers without stopping and keep track of the maximum number we have seen. Then, from the $m + 1$st number onwards, we examine the remaining numbers one at a time and we stop as soon as we see a number that is greater than the maximum of the first m numbers. If we know the underlying distribution from which these numbers were generated, we can calculate an optimal value for m that gives the highest probability of finding one of the largest numbers in the sequence.

Since we do not know the distribution of peak demand values throughout the year, we extend the distribution-agnostic stopping algorithm from [2] to our problem as shown in Figure 2; we call this algorithm "Stopping". For the first half of the year, we never call any peak days. We then set the peak demand threshold τ_D to equal the Kth-highest peak demand we have seen in the first half of the year (line 3). In the second half of the year (line 4), any days whose day-ahead peak demand forecast exceeds the threshold will be predicted as peak days (lines 5-6). In Section 4, we comment on the performance of this algorithm relative to how many days we initially skip.

3.3 The Optimization Approach

We now give an optimization algorithm based on that presented in [4], which, on any given day, uses the peak demand forecast for all the remaining days in the current year. The objective is to maximize the sum of the peak demand values of the five days that will be predicted to be peak days. In other words, this objective function aims to identify those five days which have the highest

[2]Of course, the declared number may not really be the largest number in the sequence.

1. FOR i=0 to $\lfloor \frac{N}{2} \rfloor$
2.   Do nothing
3. τ_D = Kth largest D_j for $1 \leq j \leq \lfloor \frac{N}{2} \rfloor$
4. FOR i= $\lceil \frac{N}{2} \rceil$ to N-1
5.   IF $\hat{D}_{i1} > \tau_D$
6.   Predict "tomorrow will be a peak day"

Figure 2: The Stopping Algorithm

1. $P = 0$ // how many peak days called up to now
2. FOR i=0 to N-1 // for each day of the year
3.   Recompute optimal value for τ_D
   given $5 - P$ and $\hat{D}_{i1}$ through $\hat{D}_{i,N-i}$
4.   IF $\hat{D}_{i1} > \tau_D$
5.   Predict "tomorrow will be a peak day"
6. $P = P + 1$

Figure 3: The Optimization Algorithm

forecasted peak demand, which is exactly what we want.

A high-level pseudocode of the "Optimization" algorithm is shown in Figure 3. The idea is to update the peak demand threshold τ_D every day based on the number of remaining peak days that we can call and based on the long-term forecast of the peak demand of all the remaining days in the year (line 3). The threshold update process uses finite-horizon dynamic programming and is described in detail in [4].

3.4 The Probabilistic Approach

The long-term demand forecast is not accurate, meaning that the above optimization algorithm may not work well. We now propose an algorithm that only uses the 14-day short-term forecast provided by the IESO. The idea is to compute the probability of each day being one of the top-K days out of all the days we have seen so far plus all the days for which we have a short-term forecast. If this probability exceeds a threshold τ_p that will be defined shortly, the given day will be classified as a peak day.

3.4.1 Calculating Probabilities

In order to compute probabilities over the short-term forecasts, we need to identify their distributions. According to the chi square goodness-of-fit test, we verified that the residuals of the short-term forecasts, computed as $\hat{D}_{iL} - D_i$, from 2006 till 2013 are normally distributed with a mean of zero and some standard deviation that depends on L. Thus, every short-term forecast is a random variable with a mean equal to the forecast value and a standard deviation computed from historical data. We define the following probabilities: $P(Rank_{future} = j)$, $P(Rank_{past} = j)$, and $P(Rank_{overall} = j)$.

- $P(Rank_{future} = j)$ is the probability of $\hat{D}_{i1}$ ranking jth among the 14 days for which we have a short-term forecast, i.e., among $\hat{D}_{i1}$ through $\hat{D}_{i,14}$.

- $P(Rank_{past} = j)$ is the probability of $\hat{D}_{i1}$ ranking jth compared to the peak demand on the days we have seen so far, i.e., D_1 through D_i.

- $P(Rank_{overall} = j)$ is the probability of $\hat{D}_{i1}$ ranking jth among the days we have already seen plus those for which we have a short-term forecast, i.e., D_1 through D_i and $\hat{D}_{i1}$ through $\hat{D}_{i,14}$.

Table 1: Computing ranking probabilities. Define $\theta(x, y)$ as follows: $\theta(x, y) = P(Rank_{future} = x) \times P(Rank_{past} = y)$

Final Ranking	Formula
$P(Rank_{overall} = 1)$	θ(1,1)
$P(Rank_{overall} = 2)$	θ(1,2) + θ(2,1)
$P(Rank_{overall} = 3)$	θ(1,3) + θ(2,2) + θ(3,1)
$P(Rank_{overall} = 4)$	θ(1,4) + θ(2,3) + θ(3,2) + θ(4,1)
$P(Rank_{overall} = 5)$	θ(1,5) + θ(2,4) + θ(3,3) + θ(4,2) + θ(5,1)

Assuming the short-term forecasts for different days are independent, $P(Rank_{future} = j)$ and $P(Rank_{past} = j)$ are easy to compute. For example:

$$P(Rank_{future} = 1) = \prod_{j=1}^{14} P(\hat{D}_{i1} \geq \hat{D}_{ij}) \qquad (1)$$

Since we assumed that the residuals of the short-term forecast are normally distributed, we compute $P(\hat{D}_{i1} \geq \hat{D}_{ij})$ using the probability density function for a normal distribution.

$P(Rank_{overall} = j)$ is more complex. Table 1 shows how to compute it for j between one and five. For example, $\hat{D}_{i1}$ can rank third overall under three conditions: either it ranks first in the past and third in the short-term future, or it ranks second in the past and second in the short-term future, or it ranks third in the past and first in the short-term future.

3.4.2 The Algorithm

Figure 4 gives the pseudocode for the new algorithm which we call "Probabilistic". It requires two additional input variables: initial values for τ_D and τ_p; we will discuss how to compute τ_p shortly. As in the previous algorithms, every day we check whether the day-ahead demand forecast exceeds the peak demand threshold τ_D (line 2). We also check if the day-ahead weather forecast is "extreme". We define extreme as exceeding 30 degrees Celcius. The motivation for this additional condition is to avoid false positives at the beginning of the year in case the initial threshold τ_p is too low.

If the day-ahead forecast exceeds τ_D, we compute the required probabilities (lines 4-6) and we check if tomorrow has a high probability of ranking Kth or higher. If this probability exceeds the threshold τ_p, we predict that tomorrow will be a peak day.

The last thing we need to do is to see if τ_D should be adjusted (lines 9-12). For this, we use the weather forecast again. If tomorrow is going to be a "normal weather" day, but it is still expected to exceed the demand threshold, then we should raise the threshold; specifically, we raise it to tomorrow's peak demand forecast (lines 9-10). On the other hand, if tomorrow's weather is expected to be "extreme" but the demand threshold is not going to be exceeded, then we should lower the threshold (lines 11-12).

3.4.3 Setting τ_p

In line 7, we obtain the probability that tomorrow's peak demand will be ranked Kth or higher among all the days we have seen so far plus the days for which we have a short-term demand forecast. In order to classify tomorrow as a peak day, this probability needs to exceed τ_p. To choose a value for τ_p, we use the following data-driven approach. Using the actual and estimated demand data from the previous year, we compute $P(Rank_{overall} \leq K)$ for each day in the past year. We then check this probability for the actual K peak days and choose τ_p to be the minimum of these. For example,

```
FOR i=0 to N-1 // for each day of the year
  IF D̂_i1 ≥ τ_D and extreme weather forecast
    FOR j=1 to K
      Compute P(Rank_future = j)
      Compute P(Rank_past = j)
      Compute P(Rank_overall = j) based on Table 1
    IF P(Rank_overall ≤ K) ≥ τ_p
      Predict "tomorrow will be a peak day"
  ELSE IF D̂_i1 ≥ τ_D and normal weather
    τ_D = D̂_i1
  ELSE IF D̂_i1 < τ_D and extreme weather
    τ_D = D̂_i1
```

Figure 4: The Probabilistic Algorithm.

Table 2: An example of a six-day peak demand forecast

Forecast	$\hat{D}_{iL}$	σ	P($\hat{D}_{i1}$>$\hat{D}_{iL}$)
$\hat{D}i1$	23665	210	N/A
$\hat{D}i2$	23932	584	0.368
$\hat{D}i3$	16630	666	1
$\hat{D}i4$	17635	716	1
$\hat{D}i5$	16172	804	1
$\hat{D}i6$	18158	954	1

in 2012, each one of the five peak days had $P(Rank_{overall} \leq 5)$ above 0.1, so for 2013 we can set $\tau_p = 0.1$. We will comment on the effect of τ_p on our algorithm in Section 4.

3.4.4 An Example

Let the current value of τ_D be 23,275 megawatts and assume that tomorrow's weather is expected to be extreme. Assume the demand forecasts and their standard deviations (as computed from historical data) are shown in the Table 2. To simplify the example, assume that our short-term forecast is only for the next six days, not 14. Note that the standard deviation increases as L increases.

Since we assumed that the residuals of the short-term forecast are normally distributed, we can compute P($\hat{D}_{i1}$>$\hat{D}_{iL}$) using the probability density function for a normal distribution:

$$f(x, \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \qquad (2)$$

Thus, for example, P($\hat{D}_{i1}$>$\hat{D}_{i2}$) $= f(23665 - 23932, 0, 210 + 584) = 0.368$.

Similarly, to compute the probability that tomorrow's forecast of 23,665 exceeds the actual peak demand on some day in the past, say D_i, we use $f(23665 - D_i, 0, 210)$. Since actual demand does not have a standard deviation, the standard deviation term in the probability function equals the standard deviation of the day-ahead forecast. Table 3 gives an example of the current top-five peak demand values since the beginning of the year and the corresponding probabilities with respect to tomorrow's forecast.

Based on the numbers from Table 2 and Table 3, we can compute the overall ranking probabilities as shown in Table 4. This gives $P(Rank_{overall} \leq 5) = 0 + 0.007 + 0.038 + 0.050 + 0.034 = 0.129$. Assuming $\tau_p = 0.1$, we would indeed predict that tomorrow will be a peak day.

3.5 Discussion

Table 3: An example showing five predicted peak days

Rank	D_i	P($\hat{D}_{i1}$<D)
1	24636	1
2	24107	0.9823
3	23910	0.8783
4	23801	0.7413
5	23745	0.6484

Table 4: An example of computing the ranking probabilities

Rank	$P(Rank_{future})$	$P(Rank_{past})$	$P(Rank_{overall})$
1	0.368	0	0
2	0.632	0.017	0.007
3	0	0.104	0.038
4	0	0.136	0.050
5	0	0.093	0.034
6	0	0.648	N/A

Table 5 summarizes selected features of the four algorithms discussed in this section. All but the Stopping algorithm use a dynamic threshold τ_D that is updated throughout the year; the Optimization and Probabilistic algorithms may update the threshold as often as every day, while the CPP algorithm considers updating the threshold every 2 weeks. Furthermore, our algorithm (Probabilistic) is the only one that uses the short-term forecast. Also note that the Optimization algorithm is the only one that predicts exactly K peak day, while the other algorithms may predict more of fewer peak days.

4. RESULTS

We implemented the algorithms using Matlab R2010.b and evaluated their precision, recall and running time on Ontario's demand data from 2005 till 2013. The historical load, short-term and long-term forecast data were downloaded from the IESO Website at `ieso.ca`, and we obtained the day-ahead weather forecasts for the city of Toronto from `climate.weather.gc.ca`. The evaluation was done on a Windows machine with an Intel Core i5 processor and 4 GB of RAM.

There was some data pre-processing that had to be done. The short-term forecast data had some gaps (missing forecasts) and duplicates (two or more forecasts for the same day); roughly 2 percent of the data had these problems. We filled in the gaps by interpolation and we removed duplicates by ignoring all but the latest forecasts. As a "sanity check", we compared the total demand before and after data cleaning and found that they are very similar. Note that only the Probabilistic algorithm uses the short-term forecast and is affected by our data cleaning decisions.

We gave all but our (Probabilistic) algorithm an unfair advantage: instead of providing demand forecasts as input, we give them the actual peak demand values. Our algorithm continues to use short-term forecasts. For those algorithms which need an initial value of τ_D (CPP and Probabilistic), we set it to be the maximum peak demand from the long-term forecast, assuming normal weather, minus a Load-Forecast-Uncertainty (LFU) value of 1600 megawatts. The LFU value is provided by the IESO and is related to the uncertainty of the long-term forecast. The initial τ_D and δ values used by the CPP algorithm in each year are shown in Table 6. Our algorithm uses the same initial τ_D's. The (fixed) demand thresholds calculated by the Stopping algorithm for each year are also listed in Table 6.

Table 5: Comparison of the four algorithms

Algorithm	Threshold	Required forecast	Threshold update frequency
CPP	Dynamic	Next Day	2 weeks
Stopping	Fixed	Next Day	Daily
Optimization	Dynamic	Long-term	Daily
Probabilistic	Dynamic	Short-term	Daily

Table 6: Initial parameter values

	CPP		Stopping
Year	τ_D	δ	τ_D
2005	22521	16	25126
2006	24776	16	23045
2007	24859	16	23299
2008	24822	16	22302
2009	23404	16	19044
2010	22426	16	22806
2011	22393	16	20689
2012	22355	16	22791
2013	22396	4	21385

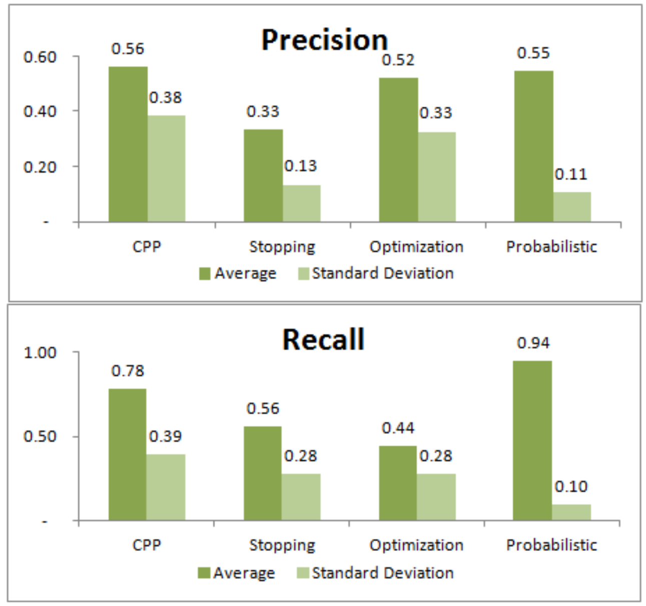

Figure 5: Average and standard deviation of the precision and recall of each algorithm

4.1 Summary of Results

Figure 5 provides an executive summary of our results. It shows the average and the standard deviation of precision and recall of each algorithm on Ontario demand data from 2005 till 2013. Our algorithm has the second-highest precision (only 0.01 behind the winner), and easily the highest recall. It has few false alarms (high precision) while still being able to identify 94 percent of the actual peak days (high recall). The standard deviation of the precision and recall of our algorithm from year to year is significantly lower than that of the other algorithms. This means that our algorithm is more consistent, while the other algorithms may have "bad years" with very low precision and recall. Interestingly, the CPP algorithm outperformed Stopping and Optimization on average, though its standard deviation was higher.

Furthermore, the number of peak days identified by our algorithm varied from 7 to 11. Thus, to identify four or five of the actual five peak days, we may need to call a total of about 10 days as peak days.

As for the running time, Probabilistic and Optimization took 6 seconds on average to make day-ahead decisions. This is slower than CPP and Stopping (under 1 second on average), but still very reasonable in a day-ahead scenario. Additionally, our algorithm incurs the overhead of setting τ_p based on historical data, but this also takes only a few seconds. Thus, our algorithm obtains high precision and recall, consistently performs well, and is fast enough that it can be used in practice.

4.2 Detailed Results

Figures 6 and 7 show the precision and recall, respectively, of each algorithm for each tested year. Note that the results for our algorithm start in 2007. This is because the short-term forecast is only available since 2006 and the Probabilistic algorithm needs one year of past data to determine the probability threshold τ_p.

The CPP algorithm computed the historical distribution of actual pea days as: one in June, three in July and one in August. Even though this algorithm had reasonably good precision and recall on average, it was inconsistent from year to year. For example, in 2005, it predicted 37 peak days, which led to perfect recall but low precision. This was largely because the initial threshold of 22521 (see Table 6) was too low. In June 2005 alone, 12 peak days were predicted before the demand threshold was raised. On the other hand, in 2008, the initial threshold was too high and zero peak days were predicted, even though the threshold was being lowered every 15 days. Thus, the precision and recall are both zero.

We experimented with larger values of δ to see if more drastic threshold adjustments would help when the initial thehesold values were too low or too high. However, even after we tripled the value of δ, the CPP algorithm still obtained zero precision and recall in 2008, and precision increased only slightly in 2005.

In general, the Stopping algorithm performed poorly because it never calls any peak days in the first half of the year (more precisely, before mid-July). Thus, it misses many peak days that happen early in the summer. We modified the Stopping algorithm to skip fewer than half the year, down to $1/e$ of the year, but precision and recall improved only slightly.

The Optimization algorithm had high precision, but the lowest recall, mainly because it is limited to predicting exactly five peak days. Thus, it was the most conservative of the four algorithms. Notably, in 2011, this algorithm did not predict any peak days, so its precision and recall that year were zero. We then allowed the Optimization algorithm to predict more than five peak days but the results did not improve. Even if it was allowed to predict up to 10 peak days, the precision and recall for 2011 were still zero.

Finally, our Probabilistic algorithm was the most consistent from year to year. Its precision did not drop below 0.4 and recall did not drop below 0.8. It had very few false negatives and relatively few false positives. The Optimization algorithm had even fewer false positives, but many more false negatives.

We also experimented with different values of τ_p for the Probabilistic algorithm and found that by raising τ_p from 10 percent (computed as described in Section 4.4.3) to 20 percent, we obtained an increase in precision but a drop in recall and an increase in the number of false negatives, i.e., we made the algorithm more conservative.

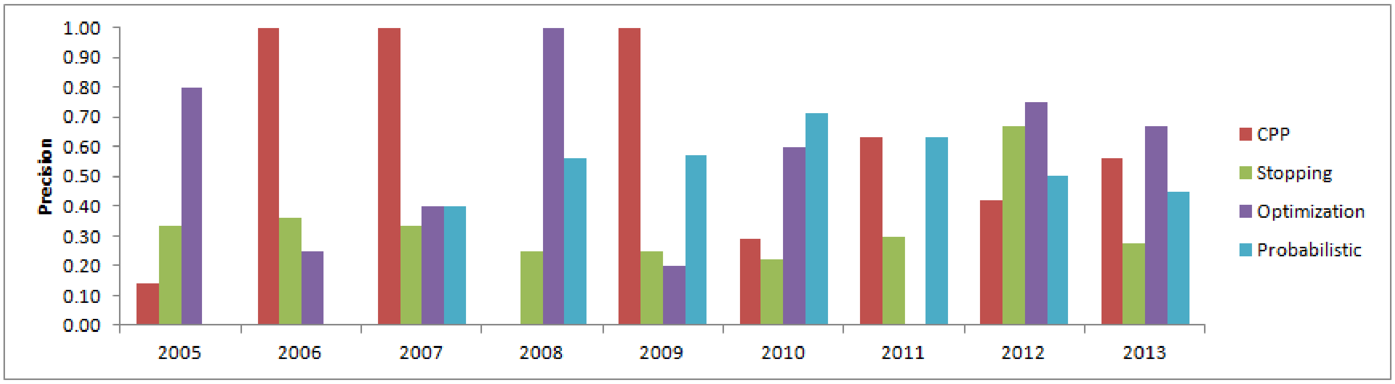

Figure 6: Precision of each algorithm from 2005 to 2013

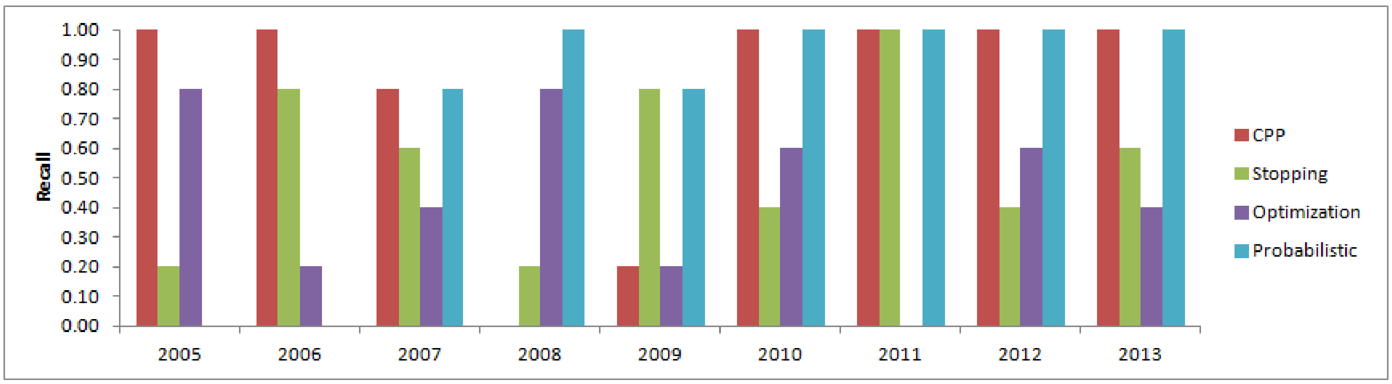

Figure 7: Recall of each algorithm from 2005 to 2013

5. CONCLUSIONS

In this paper, we presented an algorithm for making day-ahead predictions in the context of Ontario's 5CP program, regarding the likelihood of tomorrow being one of the top five peak-demand days of the year. One of the insights behind our algorithm was to only use the short-term peak demand forecast, which is more accurate than the long-term forecast. Experimental results on Ontario's demand data from 2005 till 2013 confirmed the advantages of our approach versus existing algorithms that solve related problems.

From a policy-making point of view, one observation that follows from our results is that it is very difficult to achieve 100 percent precision and recall in the 5CP setting where peak-pricing events are not broadcast by the utility, i.e., it is very difficult to correctly guess all the actual peak days in an on-line fashion without incurring any false positives. This means that consumers who want to make sure they reduce load on all five actual peak days end up having to reduce load on a few other days as well. This may impact their business operations more than intended.

We envision at least two directions for future work. One is to do further investigation on the effect of the probability threshold τ_p used by our algorithm on precision and recall. Another is to include a way for customers to specify how serious false positives and false negatives are to them, and use this information to adjust the thresholds used by the algorithm.

6. REFERENCES

[1] The 5CP program. `http://ieso-public.sharepoint.com/Pages/Participate/Settlements/Global-Adjustment-for-Class-A.aspx`

[2] M. Babaioff, N. Immorlica, and R. Kleinberg. Matroids, secretary problems, and online mechanisms. In *Symp. on Discrete Algorithms SODA 2007*, 434-443.

[3] J. Bode, C. Churchwell, S. George, and F. Sullivan. 2012 California statewide non-residential critical peak pricing evaluation. Technical report, Freeman, Sullivan & Co, 2013.

[4] W. Chen, X. Wang, J. Petersen, R. Tyagi, and J. Black. Optimal scheduling of demand response events for electric utilities. *IEEE Transactions on Smart Grid*, 4(4):2309–2319, 2013.

[5] S. De Adder. Seeking ways to reduce electricity consumption. *Western News at the Western University of Ontario*, June 20, 2012.

[6] S. Gallagher. Electric demand response tariffs clean-up. Accessed on 18 October 2013, at `http://www.pge.com/nots/rates/tariffs/tm2/pdf/ELEC_3221-E.pdf`

[7] R. Kleinberg. A multiple-choice secretary algorithm with applications to online auctions. In *Symp. on Discrete Algorithm SODA 2005*, 630-631.

[8] J. Preater. On multiple choice secretary problems. *Mathematics of Operations Research*, 19(3):597–602, 1994.

[9] J. Spears. Ontario power fee sets new record. Accessed on 18 October 2013, at `http://www.thestar.com/business/economy/2013/09/03/ontario_power_fee_sets_new_record.html`.

[10] R. H. Stockbridge and C. Zhu. A direct approach to the solution of optimal multiple-stopping problems. In *Optimization, Control, and Applications of Stochastic Systems*, 283–299. Springer, 2012.

[11] R. Tyagi, J. W. Black, and J. Petersen. Optimal scheduling of demand response events using options valuation methods. In *Power and Energy Society General Meeting, PES'11*, 1–5, 2011.

[12] X. Zhang. Optimal scheduling of critical peak pricing considering wind commitment. *IEEE Transactions on Sustainable Energy*, PP(99):1–1, 2013.