Data-Guided Approach for Learning and Improving User Experience in Computer Networks

Yanan Bao YNBAO@UCDAVIS.EDU
Xin Liu LIU@CS.UCDAVIS.EDU
Amit Pande PANDE@UCDAVIS.EDU
University of California, Davis, CA 95616, USA

Editor: Geoffrey Holmes and Tie-Yan Liu

Abstract

Machine learning algorithms have been traditionally used to understand user behavior or system performance. In computer networks, with a subset of input features as controllable network parameters, we envision developing a data-driven network resource allocation framework that can optimize user experience. In particular, we explore how to leverage a classifier learned from training instances to optimally guide network resource allocation to improve the overall performance on test instances. Based on logistic regression, we propose an optimal resource allocation algorithm, as well as heuristics with low-complexity. We evaluate the performance of the proposed algorithms using a synthetic Gaussian dataset, a real world dataset on video streaming over throttled networks, and a tier-one cellular operator's customer complaint traces. The evaluation demonstrates the effectiveness of the proposed algorithms; e.g., the optimal algorithm can have a 400% improvement compared with the baseline.

Keywords: Classification, Resource Allocation, User Experience

1. Introduction

The growth of storage and processing capability allows the collection and processing of massive and continuous data streams that are generated by modern computer systems such as data centers and mobile carriers. Large scale data collection and analysis enables the extraction of useful information and knowledge about end-user experience that was previously unavailable. Data-driven models have been used in computer networks to characterize or understand the network performance, application performance or user behavior at large. Balachandran et al. (2013) uses machine learning to model user-engagement in watching videos over internet and to understand the impact of network parameters on such user engagement. Baik et al. (2015) models the end-user video watching experience based on the buffering or bitrate variations in the "distorted" video, which can be tracked back to the network using results in Chan et al. (2012). Similarly, Huang et al. (2010) perform a reliability analysis of nodes in a wireless sensor network using logistic regression, a widely used machine learning approach. The wide-spread availability of such logged/stored data has extended to other application domain such as smart homes and smart cities. However, most research only focused on passive measurements and analysis of such datasets using machine-learning techniques.

The generated prediction models can estimate the end-user's Quality of Experience (QoE), hence allowing network operators to observe the network status and to take suitable "action" to improve its performance. Ideally, it would be desirable that the model can guide the appropriate allocation of network resources. For example, in a typical example of video being streamed by a cellular service provider to dozens of users in a cell, the operator can take the real-time feedback of received video quality to allocate bandwidth or any other resource at disposal to improve user experience.

In this work, we articulate a framework to enable such applications. Generally speaking, machine learning returns a classifier based on training data, and it is used to predict the labels of test data: positive and negative (users). With some network resources at our disposal, we could increase the overall user satisfaction by transforming likely positives into likely negatives – one intuitive strategy is to evenly distribute the available additional resources to all predicted unsatisfied users (positives). However, we argue that the learned classifier (both the model and its parameters) provides more information than just the predicted labels, and thus we should use it as the objective function in modeling the resource optimization problem. Such integration allows us to better guide the allocation of network resources for optimal performance improvement. This paper makes the following main contributions:

1. This paper investigates the use of machine learning techniques (particularly logistic regression) to improve network resource allocation with the goal of maximizing the total number of satisfied users.
2. Given the logistic regression-based resource allocation problem, we propose a Sweep algorithm that can find the optimal solution, as well as a heuristic algorithm with low complexity.
3. The algorithms are evaluated on both synthetic and real world datasets from computer networks. Results indicate the Sweep algorithm can outperform the baseline up to 400%.

The remainder of this paper is organized as follows. Sec. 2 discusses the related work. In Sec. 3, the framework of data-driven resource allocation is introduced. Sec. 4 presents an algorithm to find the optimal solution for logistic regression-based resource allocation, as well as a heuristic algorithm with low complexity. Three experiments are reported in Sec. 5, the first is designed to have ground truth and the others come from the real world. Finally, Sec. 6 discusses the limitations and future work and Sec. 7 concludes the paper.

2. Related Work

The classification model considered in the paper is logistic regression (Hosmer Jr and Lemeshow, 2004; Harrell, 2001). We choose it because it is one of the most widely used classification models with good intuitions and explainable results.

A large literature considers the learning-based cost-efficient decision making. Horvitz and Mitchell (2010) discuss the pipeline of data collection, predictive model, and decision analysis. The problem of patient readmission in hospitals with congestive heart failure is considered by Bayati et al. (2014). The authors construct a classifier to predict readmissions and propose to use patient-specific interventions to reduce the cost. The combination of prediction and allocating interventions shows a reduction of both rehospitalization rate and cost. In the problem of machine learning and traveling repairman, the combination of learning and decision making is studied by Tulabandhula and Rudin (2013) and Tulabandhula and Rudin (2014). A recommendation system is designed by Qu et al. (2014) to maximize taxi drivers' profits in a cost-efficient way. However, none of the existing work, to the best of our knowledge, have considered data-driven network resource allocation problem.

Resource allocation has been widely studied in different areas. Extensive research can be found on the power, rate and bandwidth allocations in wireless networks, with the water-filling algorithm to be the most well known (Song and Li, 2005; Shen et al., 2005). Similarly,

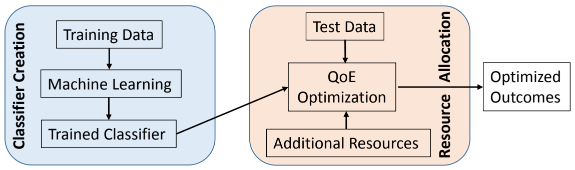


Figure 1: The framework of data-driven resource allocation.

the power management, speed-scaling and capacity provisioning have been considered in the design of server farms (Gandhi et al., 2010; Buyya et al., 2010).

The combination of learning and resource allocation is discussed by Lee et al. (2010) and Berral et al. (2010) in their specific situations. However, to the best of our knowledge, none of these papers applied the classifier learned on training data to guide resource allocation on test data.

3. Problem Formulation

Fig. 1 illustrates the overall framework for the data-driven network resource allocation. First of all, we choose an appropriate classifier based on the types of resources available and the nature of the problem. Then, we train the parameters of the classifier based on the training data. Last, the trained classifier is fed into the resource allocation module as the objective function of allocating resource to test data.

3.1. Classifier Creation

Consider instances in a D-dimensional space. Given a training set including N instances, the features of instance i are denoted by a vector $\boldsymbol{x_i} = [x_{i,1}, x_{i,2}, ..., x_{i,D}]^T$, and the corresponding label is y_i, for $i = 1, 2, ..., N$. In this work, we consider two labels: positive (label 1) and negative (label 0).

A classifier $\hat{y}_i = f(\boldsymbol{x_i}, \boldsymbol{w})$ maps the instance $\boldsymbol{x_i}$ to the estimated label $\hat{y}_i$, and the parameter $\boldsymbol{w}$ can be learned from the training data. The difference between y_i and $\hat{y}_i$ is depicted by a function named $\text{Loss}(y_i, \hat{y}_i)$. Based on the training data, the optimal $\boldsymbol{w}^*$ is found by solving the following problem

$$\arg\min_{\boldsymbol{w}} \sum_{i=1}^{N} \text{Loss}(y_i, f(\boldsymbol{x_i}, \boldsymbol{w})). \tag{1}$$

For example, linear regression, logistic regression, and Support Vector Machine (SVM) apply L2-norm distance, sigmoid function, and hinge function as their loss functions, respectively.

3.2. Resource Allocation

The goal of the resource allocation is to improve user experience on test data by allocating resource to maximize the expected number of satisfied users. We consider maximize the total number of satisfied users because it is directly related to the revenue of the service providers. For example, the unsatisfied complaints received by cellular operators is the key performance indicator of their services. With this goal, the fairness among users is not guaranteed, and we will consider the fairness in our future work. Specifically, based on the classifier and its trained parameters, one can predict the labels of the test data.

(Following the tradition, we assume that negatives represent users in normal states and positives represents troubled users.) Based on the prediction, one can take actions to ameliorate the likely positives. While such actions may depend on domain knowledge, allocating additional resource to them can be a general solution.

We focus on the application scenarios, specifically computer networks, where one can allocate (additional) resources to improve the experience of users. Such resources may include network bandwidth, time, transmission power, monetary compensation, etc.. In such scenarios, we assume that the values of a subset of features can be affected when allocated additional resource and thus user status can be improved. We note that our model does not apply to the following cases: 1) the values of features cannot be modified; e.g., the age of a person, the maker of a vehicle, the weather of a day; 2) labels are the causes and features are the effects. For example, a cold usually leads to an increase of white blood cells, but it may not help the patient recover by simply trying to decrease the number of white blood cells.

The objective is to minimize the number of positives on the test data by allocating K types of resource denoted by $\boldsymbol{R} = [R_1, R_2, ..., R_K]^T$. We assume that when $\boldsymbol{r_i} = [r_{i,1}, r_{i,2}, ..., r_{i,K}]^T$ amount of resource is given to $\boldsymbol{x_i}$, the values of its features will change to $\boldsymbol{g}(\boldsymbol{x_i}, \boldsymbol{r_i})$, for $i = 1, 2, ..., M$, and the classier gives the its estimated probability to be positive as $f\left(\boldsymbol{g}\left(\boldsymbol{x_i}, \boldsymbol{r_i}\right), \boldsymbol{w}^*\right)$. Therefore we have the following optimization problem.

$$\min_{\boldsymbol{r_1}, \boldsymbol{r_2}, ..., \boldsymbol{r_M}} \quad \sum_{i=1}^{M} f\left(\boldsymbol{g}\left(\boldsymbol{x_i}, \boldsymbol{r_i}\right), \boldsymbol{w}^*\right) \tag{2}$$

$$s.t. \quad \sum_{i=1}^{M} \boldsymbol{r_i} \leq \boldsymbol{R}; \tag{3}$$

$$\boldsymbol{r_i} \geq \boldsymbol{0}, \quad i = 1, 2, ..., M. \tag{4}$$

Here the inequality symbol represents component-wise inequality.

3.3. Logistic Regression-Based Resource Allocation

In this paper, we use logistic regression as the classifier to study the framework in details. With logistic regression, the training process can be modeled as

$$\arg\max_{\boldsymbol{w}} \ \sum_{i=1}^{N} y_i \log\left(\frac{1}{1+\exp(-\boldsymbol{w}^T\boldsymbol{x_i})}\right) + (1-y_i)\log\left(\frac{1}{1+\exp(\boldsymbol{w}^T\boldsymbol{x_i})}\right) + |\boldsymbol{w}|, \tag{5}$$

where $y_i = 0$ or 1, for $i = 1, 2, ..., N$, $\boldsymbol{w} = [w_1, w_2, ..., w_D]^T$, and L_1 regularization is applied.

Assume the changes of features have linear relation with the resources allocated, i.e., $\boldsymbol{g}(\boldsymbol{x_i}, \boldsymbol{r_i}) = \boldsymbol{x_i} + \mathbb{Q}\boldsymbol{r_i}$, where $\mathbb{Q}$ is a $D \times K$ matrix, and $\mathbb{Q}_{d,k}$ is the effect of resource k on feature d. Based on the trained logistic regression classifier, when instance i is allocate with $\boldsymbol{r_i}$ amount of resources, its probability to be positive is $p_i = \frac{1}{1+\exp(-\boldsymbol{w}^T\boldsymbol{x_i} - \boldsymbol{w}^T\mathbb{Q}\boldsymbol{r_i})}$. Denote $-\boldsymbol{w}^T\boldsymbol{x_i}$ by C_i, and $\mathbb{Q}^T\boldsymbol{w}$ by $\boldsymbol{a} = [a_1, a_2, ..., a_K]^T$. Here a_k $(k = 1, 2, ..., K)$ represents the aggregated effects of resource k on the linear combination. For the ease of explanation, we assume $a_k \geq 0$ $(k = 1, 2, ..., K)$, i.e., allocating any of the K types of resources decreases instance i's probability to be positive. Therefore, the expected total number of positives is $\sum_{i=1}^{M} p_i = \sum_{i=1}^{M} \frac{1}{1+\exp(C_i + \boldsymbol{a}^T\boldsymbol{r_i})}$. The optimization problem minimizing the expected total number of positives is

$$\textbf{(P-1)} \min_{\boldsymbol{r_1}, \boldsymbol{r_2} \cdots \boldsymbol{r_M}} \quad \sum_{i=1}^{M} \frac{1}{1+\exp(C_i + \boldsymbol{a}^T\boldsymbol{r_i})}; \tag{6}$$

$$s.t. \quad \sum_{i=1}^{M} \boldsymbol{r_i} \leq \boldsymbol{R}; \tag{7}$$

$$\boldsymbol{r_i} \geq \boldsymbol{0}, \quad i = 1, 2, ..., M. \tag{8}$$

This is a non-convex optimization, which means it cannot be solved efficiently by gradient-based numerical methods. Next, we propose an algorithm to obtain an optimal solution.

4. Resource Allocation Algorithms

In the section, we first propose an optimal algorithm to **(P-1)**. We then propose a heuristic meta algorithm with low complexity based on the insights of the optimal algorithm.

4.1. The Optimal Resource Allocation Algorithm

First, we derive an equivalent optimization problem that simplifies the formulation in **(P-1)** as follows

$$\textbf{(P-2)} \min_{s_1,s_2,...,s_M} \quad \boldsymbol{F}([s_1,s_2,...,s_M]) = \sum_{i=1}^{M} \frac{1}{1+\exp(C_i+s_i)}; \tag{9}$$

$$s.t. \quad \sum_{i=1}^{M} s_i \le S; \tag{10}$$

$$s_i \ge 0, \quad i=1,2,...,M; \tag{11}$$

where $S = \boldsymbol{a}^T\boldsymbol{R}$. Here s_i stands for equivalent resource, which is a weighted summation of multiple types of resources, and the effectiveness of different resource is reflected by the weights. In this problem, the number of variables is reduced from MD to M, and the number of constraints is reduced from $D(M+1)$ to $M+1$. We have the following proposition.

Proposition 1 *(P-1) and (P-2) are equivalent problems.*

Proof Assume the optimal solution and minimum for **(P-1)** and **(P-2)** are $([\boldsymbol{r_1}^*, \boldsymbol{r_2}^*, ..., \boldsymbol{r_M}^*], v_1^*)$ and $([s_1{}^*, s_2{}^*, ..., s_M{}^*], v_2^*)$, respectively. Define

$$s_i = \boldsymbol{a}^T\boldsymbol{r_i}^*, \quad i=1,2,...,M. \tag{12}$$

Since $\boldsymbol{a} \ge \boldsymbol{0}$, $[s_1,s_2,...,s_M]$ meet constraints (10) and (11). Therefore $[s_1,s_2,...,s_M]$ is in the feasible set of **(P-2)** and we have $v_2{}^* \le v_1{}^*$.

Define

$$r_{i,k} = \frac{s_i{}^* R_k}{\boldsymbol{a}^T\boldsymbol{R}} \text{ for } i=1,2,...,M \text{ and } k=1,2,...,D, \tag{13}$$

which meets constraint (8). Moreover, since

$$\sum_{i=1}^{M} r_{i,k} = \frac{R_k \sum_{i=1}^{M} s_i{}^*}{\boldsymbol{a}^T\boldsymbol{R}} \le R_k, \text{ for } k=1,2,...D, \tag{14}$$

we have constraint (7) satisfied. Therefore $[\boldsymbol{r_1}, \boldsymbol{r_2}, ..., \boldsymbol{r_M}]$ is in the feasible set of **(P-1)** and we have $v_1{}^* \le v_2{}^*$.

Therefore, $v_1{}^* = v_2{}^*$. ■

Even though the equivalent problem is still a non-convex optimization, we find special properties that one of the optimal solutions has to satisfy, given in Lemma 3 and 4 in Appendix A. These properties reduce the searching space of the optimal solutions. Lemma 4 shows one optimal solution allocates the equivalent resource to a contiguous set of instances sorted based on C_i for $i=1,2,...,M$. Therefore we can test all contiguous sets to find the right set of candidates. After deciding a set of candidates to receive resources, the equivalent resource has to be allocated to the candidates in a water filling way (Lemma 3). Based on these properties, Algorithm 1 is designed to find an optimal solution $[s_1^*, s_2^*, ..., s_M^*]$. Then Eq. (13) is used to calculate resource $[\boldsymbol{r_1}^*, \boldsymbol{r_2}^*, ..., \boldsymbol{r_M}^*]$ in the original dimensions.

Algorithm 1: Sweep Algorithm

```
Data: $C_i$ for $i = 1, ..., M$, equivalent resource: $S$
Result: optimal resource allocation solution: $[s_1^*, s_2^*, ..., s_M^*]$
1  Sort $C_i$ in ascending order and denote the sorted list by $[C_1^s, C_2^s, ..., C_M^s]$;
2  Initialize $s_i^* = \frac{S}{M}$, for $i = 1, 2, ..., M$, which stands for the best solution to date;
3  for each contiguous subset of $[C_1^s, C_2^s, ..., C_M^s]$, denoted by $[C_l^s, ..., C_r^s]$;
4  %% $l$ and $r$ are the most left index and the most right index, respectively%% do
5      if $\sum_{j=l}^{r-1}(C_r^s - C_j^s) \leq S$ then
6          $C^* = C_r^s + \frac{S - \sum_{j=l}^{r-1}\left(C_r^s - C_j^s\right)}{r-l+1}$;
7          $s_j = C^* - C_j^s$ for $j = l, ..., r$;
8          $s_j = 0$ for $j = 1, ..., l-1$ and $j = r+1, ..., N$;
9          if $\mathbf{F}([s_1, s_2, ..., s_M]) < \mathbf{F}([s_1^*, s_2^*, ..., s_M^*])$ then
10             $[s_1^*, s_2^*, ..., s_M^*] = [s_1, s_2, ..., s_M]$;
11         end
12     end
13 end
```

This Sweep algorithm first sorts C_i in ascending order, and the sorted list is denoted by $[C_1^s, C_2^s, ..., C_M^s]$ (line 1). Then it initializes a naive solution that allocates the equivalent resource S evenly to M instances as the best solution to date (line 2). Note that the initial solution can be any solution that is feasible. After these two steps, the algorithm starts to update the best solution by trying different ways to allocate resource (the for loop). In each trial, a contiguous subset of $[C_1^s, C_2^s, ..., C_M^s]$ denoted by $[C_l^s, ..., C_r^s]$ is selected, and S is allocated to the instances in the subset in a water filling way. In particular, first the algorithm assesses if the resource is enough to achieve the minimal water level for all the instances in the subset: If the existing resource is insufficient, this trial is skipped (line 5); Otherwise, the achieved water level is denoted by C^* (line 6). The instances in the subset are allocated with resource according to this water level (line 7); And the instances out of the subset are not allocated with any resource (line 8). Then, this trial is compared with the best solution to date, and the best solution is updated if the trial achieves a lower objective function (line 9 to line 11). After all the contiguous subsets are tested, the best solution is returned.

Proposition 2 *The Sweep algorithm obtains an optimal solution.*

Proof is given in Appendix A.

The complexity of this algorithm is $O(M^3)$. Specifically, sorting has a complexity of $O(M \log(M))$. There are $O(M^2)$ combinations of the left boundaries and right boundaries, and in each combination, water filling from the left boundary to the right boundary on average has a complexity of $O(M)$.

4.2. Algorithms with Low Complexity

Due to its $O(M^3)$ complexity, the Sweep algorithm does not scale to a large test data. For example, consider 10,000 instances in the test data. It takes a PC with a 3.20GHz CPU and 8 GB memory several hours to find the optimal solution. However there are scenarios that have a large test data and sensitive delay, e.g., a global data center with hundreds of thousands of concurrent users. Therefore, we consider a meta algorithm that joins three heuristic algorithms with low complexity: Average, Water Filling and Binary Search. The Average algorithm allocates resource uniformly to the instances predicted as positives. This is also the baseline we can use to evaluate the performance of our optimal algorithm. The Water Filling algorithm and the Binary Search algorithm are illustrated as Algorithm 2 and Algorithm 3, respectively. Both algorithms allocate resource to the instances with small $\eta'(C_i)$-s first and they both will achieve a water level for the instances with resource allocation[1]. As defined by Eq. (21) in Appendix A, $\eta'(x) = \frac{-\exp(x)}{(1+\exp(x))^2}$.

Algorithm 2: Water Filling

```
Data: $C_i$ for $i = 1, ..., M$, equivalent resource: $S$
Result: resource allocation results: $[s_1, s_2, ..., s_M]$
Sort $\eta'(C_i)$-s in ascending order and denote the indices by $I$; Rank $C_i$ according to $I$ and denote the new list by $[C^s_1, C^s_2, ..., C^s_M]$;
$C^+_i = C^s_i$ if $C^s_i \geq 0$; otherwise $C^+_i = -C^s_i$, for $i = 1, 2, ..., M$;
$C^* = 0$; $S_{\text{remain}} = S$; $N_{\text{below}} = 0$; $S_{\text{last}} = 0$;
Initialize $s_i = 0$, for $i = 1, 2, ..., M$;
for $i \leftarrow 1$ to $M$ do
    if $S_{remain} < 0$ then
        $C^* = \frac{S_{\text{remain}}}{N_{\text{below}}} + C^*$; $S_{\text{remain}} = 0$;
        break;
    end
    $S_{\text{remain}} = S_{\text{remain}} - (C^+_i - C^*)N_{\text{below}}$;
    if $C^s_i < 0$ then
        if $S_{remain} < C^* + C^+_i$ then
            $S_{\text{last}} = S_{\text{remain}}$; $S_{\text{remain}} = 0$;
            break;
        end
        $S_{\text{remain}} = S_{\text{remain}} - (C^* + C^+_i)$
    end
    $N_{\text{below}} = N_{\text{below}} + 1$; $C^* = C^+_i$;
end
if $S_{remain} > 0$ then
    $s_j = C^* - C^s_j + \frac{S_{\text{remain}}}{M}$, for $j = 1, 2, ..., M$;
else
    $s_j = C^* - C^s_j$, for $j = 1, 2, ..., i - 1$;
    $s_i = S_{\text{last}}$;
end
```

Algorithm 3: Binary Search

```
Data: $C_i$ for $i = 1, ..., M$, equivalent resource: $S$
Result: resource allocation results: $[s_1, s_2, ..., s_M]$
Sort $\eta'(C_i)$-s in ascending order and denote the indices by $I$; Rank $C_i$ according to $I$ and denote the new list by $[C^s_1, C^s_2, ..., C^s_M]$;
$i_{\text{min}} = 1$; $i_{\text{max}} = M$;
Initialize $s_i = 0$, for $i = 1, 2, ..., M$;
while $i_{min} \leq i_{max}$ do
    $i_{\text{mid}} = \frac{i_{\text{min}} + i_{\text{max}}}{2}$;
    $C_{\text{max}} = \max([C^s_1, C^s_1, ..., C^s_{i_{\text{mid}}}])$;
    $S_{\text{need}} = \sum_{i=1}^{i_{\text{mid}}} \left(C_{\text{max}} - C^s_{i_{\text{mid}}}\right)$;
    if $S_{need} > S$ then
        $i_{\text{max}} = i_{\text{mid}} - 1$;
        continue;
    end
    $C^* = C_{\text{max}} + \frac{S - S_{\text{need}}}{i_{\text{mid}}}$ ;
    if $i_{mid} \leq M - 1$ then
        if $\eta'(C^*) > \eta'(C^s_{(i_{mid}+1)})$ then
            $i_{\text{min}} = i_{\text{mid}} + 1$;
            continue;
        end
    end
    break;
end
$s_i = C^* - C^s_i$, for $i = 1, 2, ..., i_{\text{mid}}$;
```

Specifically, the Water Filling algorithm increases the water level step by step until all the resource is used up. It maintains a variable S_{remain} which denotes the remaining resource after each step. In each step, a new instance is included in the set with resource allocation and the water level $\eta'(C^*)$ is updated. Note that different from the Sweep algorithm, the target of water filling in this algorithm is to achieve the same $\eta'(C_i)$, rather than the same C_i. When S_{remain} is used up, the algorithm returns the last resource allocation result.

The Binary Search algorithm searches for the right size of a subset of instances, such that allocating resource to them can meet the Karush-Kuhn-Tucker (KKT) conditions. This subset includes the instances from index 1 to i_{mid}, where the indices are based on sorting $\eta'(C_i)$-s in ascending order. During resource allocation, water filling is also used but the target is to achieve the same C_i. i_{mid} is updated by binary searching and it stops when KKT conditions (shown by Eq.(16–20) in Appendix A) are satisfied.

Both the Average algorithm and the Water Filling algorithm have complexity of O(M). The Binary Search algorithm has complexity of O($M \log(M)$) at the worst case.

Given different amount of resource, none of the three algorithms can always be the winner to minimize the number of positives. Since they all have low complexity, it is

1. The last instance that is allocated resource by the Water Filling algorithm may not be able to achieve the water level.

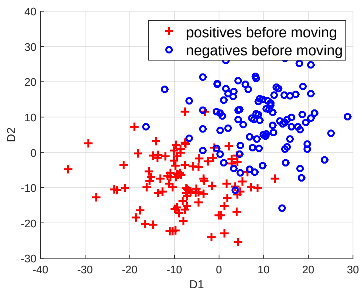
necessary to have a heuristic meta algorithm that compares them and selects the best under different available resource.

5. Experiments

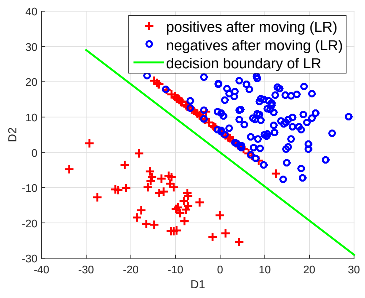
In this section, we conduct three experiments to evaluate the performance of the proposed algorithms. The first experiment is based on synthetic data generated following a 2-D Gaussian distribution. The other two experiments use real-world network user experience data: One is to minimize the subjectively unsatisfied viewers in a video transmission system; And the other is to reduce the expected number of user complaints in a cellular network. We illustrate the performance of the Sweep algorithm, and the individual performance of the three modules of the heuristic meta algorithm. To avoid crowding the figures, we do not explicitly plot the performance of the meta algorithm, which is the best of the three heuristic modules. The performance metric is the expected number of positives post resource allocation.

5.1. Gaussian Distributed Data in 2D space

The most accurate way to evaluate the performance of resource allocation is based on the ground truth, which is typically unknown in real dataset. Therefore this experiment applies a synthetic data in low dimensional space with ground truth. The synthetic data can also illustrate the intuition of the optimal algorithm clearly.



(a) Before Resource Allocation.



(b) After Resource Allocation.

Figure 2: The positives and negatives in Experiment 1.

Positives/negatives are assumed to have a 2D Gaussian distribution where the mean is (-10,-10)/(10,10) and the covariance matrix is [8, 0; 0, 8]. We consider a balanced dataset which includes an equal number of positives and negatives. Therefore, for a instance at location (x_1, x_2), its probability to be positive is $p(x_1, x_2) = \frac{d_p(x_1,x_2)}{d_p(x_1,x_2)+d_n(x_1,x_2)}$. This is the ground truth we assume. The training set contains 1,000 positives and 1,000 negatives. The test set contains 100 positives and 100 negatives, as shown in Fig. 2(a).

In this experiment, we assume only one resource exists, and it can improve feature 2 which corresponds to the y-axis. The result of the Sweep algorithm is shown in Fig. 2(b). In the figure, the solid diagonal line is the decision boundary corresponding to $\boldsymbol{w}^T\boldsymbol{x} = 0$ in Eq. (5). It is interesting to see that the Sweep algorithm pushes all instances (positives and negatives) in an asymmetric region near the decision boundary to a "virtual" diagonal line in parallel with the decision boundary. The C_i-s of these instances are contiguous when ranked, as illustrated by line 1 of the Sweep algorithm. In addition, they are on the steep slope range of the s-shaped sigmoid function (of logistic regression). Therefore, when allocated resource, their likelihood to be positive change dramatically than instances in other areas of the sigmoid function (e.g., the positives far from the decision boundary). On the other hand, not illustrated in this figure, when the amount of available resource increases, it might be optimal to allocate enough resource to positives that are far from the decision boundary because of the non-convexity of the sigmoid function. While the Sweep algorithm explicitly considers such situations, the heuristics does not, and thus explains the poor performance of the Water Filling algorithm when the available resource is abundant, as shown next.

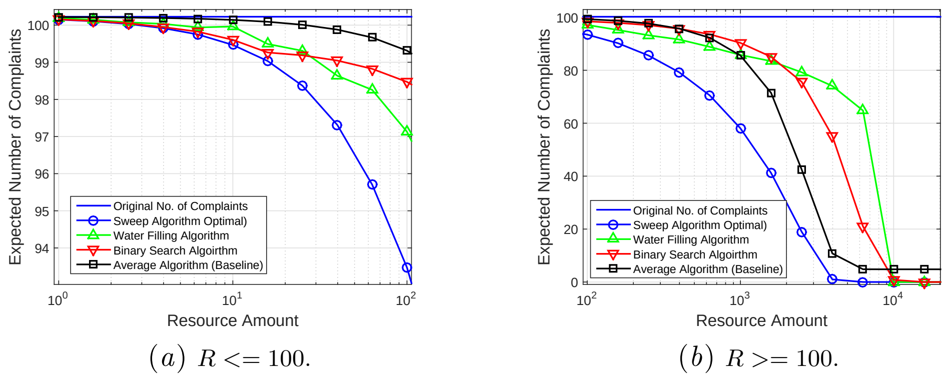


Figure 3: The impact of available resource on the expected number of positives in Experiment 1.

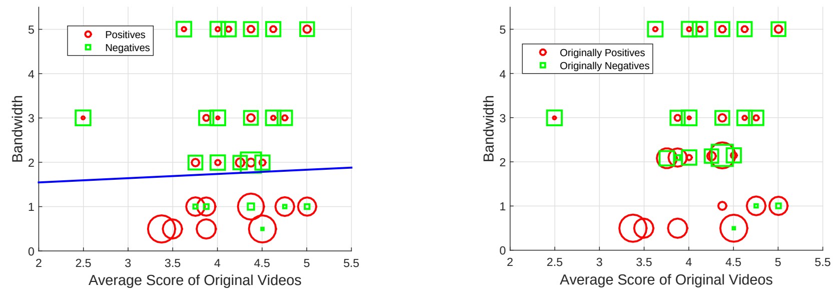
Fig. 3 shows the performance of different algorithms as a function of the total available resource. The Sweep algorithm always minimizes the total number of expected positives. The three modules of the meta algorithm achieve good performance under different ranges of available resources. In particular, the Binary Search algorithm achieves good performance when the total amount of resource is very small. The Average algorithm performs well when the resource is moderate to large. With the increase of available resource, the decreasing speeds of the curves slow down due to the fact that the remaining unsolved positives are usually further away from the decision boundary.

5.2. Video MOS Data

The Mean Opinion Score (MOS, an integer between 1 and 5) is commonly used as the subjective measurement of user experience in video transmission systems. Intuitively, network bandwidth allocated to a user is a crucial factor in determining the user's experience on videos streamed over the network, as also indicated by related research work (Mok et al., 2011; Balachandran et al., 2013; Krishnan and Sitaraman, 2013; Chan et al., 2012).

In this experiment, we use a dataset from Paudyal et al. (2014), where network bandwidth is one of the input features to model user QoE. The dataset contains 8 original videos and 8*5 streamed videos. The streamed videos are the original videos transmitted over a given network bandwidth. Each video is viewed and subjectively scored by 30 viewers. Therefore the dataset includes 240 views of the original videos and 1200 views of the streamed videos. For each view, it has two features: the bandwidth used to transmit the video and the viewer's strictness. The bandwidth has 5 discrete values: 0.5, 1, 2, 3 and 5Mb/s. The viewer's strictness is denoted by the average score the viewer gives to the original videos. Most of viewers have the average score between 3.5 and 5.

In the training data, as the common practice in video quality assessment, we assume that when MOS < 3, the viewer is unsatisfied with the viewed video and it is a positive instance. Otherwise, the viewer is satisfied and it is a negative instance. 50% of the data is used for training the classifier while the other 50% is used for testing the proposed resource allocation algorithms.

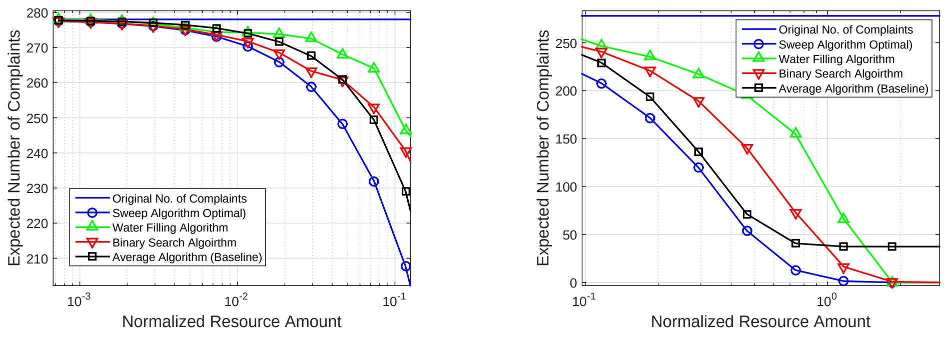


(*a*) Before Resource Allocation. (*b*) After Resource Allocation.

Figure 4: The positives and negatives in Experiment 2.

Fig. 4(*a*) illustrates the locations of positives and negatives of the test data in a 2D space. Since both features can only take several discrete values, the points are highly clustered. Therefore, we use the area of a circle/square to indicate the number of positives/negatives at the corresponding location. Note that the positives and negatives cannot be separated perfectly: The videos with the same bandwidth can be labeled diversely by the viewers with the same strictness; Even the same streamed video can get opposite labels by the viewers with the same strictness. The solid blue line corresponds to the decision boundary learned on the training data by a logistic regression model. The slope of the line shows the bandwidth plays an important role in determining viewers' experience. 10-cross validation shows the logistic regression has an accuracy of 92%.

The total bandwidth consumed by the original testing videos is 1360Mb/s. Fig. 4(*b*) shows the result on test data after allocating an additional bandwidth of 200Mb/s. The original labels of positives and negatives are still used in this figure, even though the positives may have changed to negatives after additional bandwidth allocation. Similar to Experiment 1, only the users near the decision boundary are allocated with additional bandwidth and thus are pushed upward. The instances that have either very low or very high probability to be positive are not allocated with any resource.



(*a*) Normalized $R <= 0.1$. (*b*) Normalized $R >= 0.1$.

Figure 5: The impact of available resource on the expected number of positives in Experiment 2.

Fig. 5 illustrates the impact of available resource on the expected number of positives using different algorithms. The amount of available resource is normalized by 1360Mb/s, the total bandwidth consumed by the original testing videos. When 5% additional resource is available, the optimal algorithm outperforms the baseline by 60%. Furthermore, using the optimal algorithm, increasing the existing bandwidth by 1/3 reduces 2/3 unsatisfied users. Last, the meta algorithm is a combination of the Average algorithm and the Binary Search algorithm, since the Water Filling algorithm is always dominated by one of them.

5.3. Cellular Customer Complaint Data

Cellular operators may receive complaints from customers stating that the data service cannot meet their expectation. A tier 1 operator in China records such customer complaints with time stamp and customer ID. At the same time, its network monitoring system records 13 network features for each customer on an hourly basis. Table 1 gives the features and their sample values. The operator would like to predict the likelihood of customer complaints, and then proactively allocate available resource to the potentially complaining customers to improve their performance so that the provider will receive less complaints.

Table 1: Features of the data in Experiment 3.

No.	Feature Name	Meaning	Sample Value	Weight of LR
0	Intercept	w_0 in logistic regression	1	-1.88
1	PDPSucc(%)	PDP success ratio	0∼100	-2.02
2	Attsucc(%)	Attachment success ratio	0∼100	-2.24
3	RAUSucc(%)	RAU success ratio	0∼100	-0.905
4	SessSucc	Successful sessions	15	-0.695
5	SessReq	Requested sessions	15	-2.34
6	attempt	Connection attempts	2	0.42
7	Radiostatus	Abnormal line drops	0	0.102
8	CoreFail(%)	Failures from core networks	0∼100	0.182
9	RadioFail(%)	Failures from core networks	0∼100	0.237
10	TransSucc(%)	Transmission success ratio	0∼100	-1.11
11	ThroughputD	Downlink throughput	15.26	**-0.0596**
12	TrafficD(KB)	Downlink traffic amount	7.45	2.92
13	TrafficU(KB)	Uplink Traffic amount	3.69	0.652

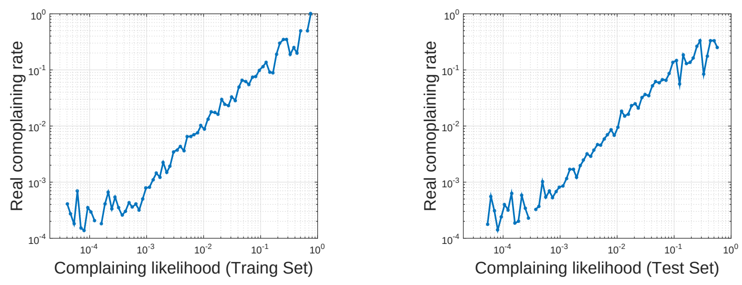
Table 2: AUC scores of machine learning methods.

Method	Logistic Regression	Neural Network	SVM	Decision Tree	Random Forest	KNN
AUC score	0.910	0.890	0.626	0.775	0.837	0.627

The dataset contains highly imbalanced data: there are 569170 negative instances and 1275 positive instances. A positive instance means the customer has made at least one complaint during the hour, and a negative instance otherwise. In the preprocessing step, each feature is normalized by its mean value. 50% of the data is used for training the classifier while the other 50% is used for testing.

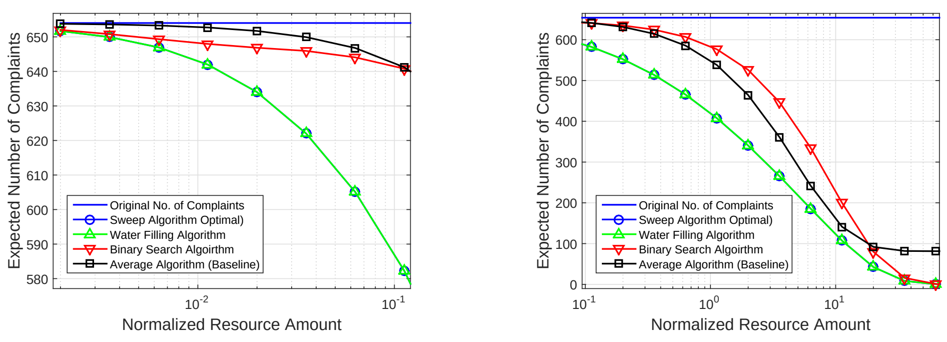
We build different classification models, including Support Vector Machine (SVM), neural networks, decision tree, random forest, logistic regression, naive Bayes, k-nearest neighbors and evaluate their performance by Receiver Operating Characteristic (ROC) curve. Logistic regression achieves the highest Area Under Curve (AUC) score as shown in Table 2. The classifier returned by logistic regression has coefficients shown in Table 1. Feature 0 is constant with value 1, which corresponds to the intercept. Features 1∼10 are related to the connection performance and stability of the networks, which depend on the implementation of network hardware and software. Feature 11 is the throughput of a user, which can be increased by allocating more bandwidth to the user. Features 12 and 13 depend on the user requirements and are not controlled by the network.

For each training instance or test instance, the classifier outputs an estimated complaining likelihood to indicate the probability that the instance is positive. Fig. $6(a)$ and Fig. $6(b)$ show that given a certain estimated complaining likelihood returned by logistic



(a) Training Set. (b) Test Set.

Figure 6: Real complaining rates vs. estimated complaining likelihoods.



(a) Normalized $R <= 0.1$. (b) Normalized $R >= 0.1$.

Figure 7: The impact of available resource on the expected number of positives in Experiment 3.

regression, what the real complaining rate is. On both the training data and test data, the larger the estimated complaining likelihood is, the higher the real complaining rate will be. The diagonal lines with angle 45° illustrate that the estimated complaining likelihood is a good match with the real complaining probability. The fluctuations on the left part of the lines are due to the smaller number of samples in that region.

In testing, each 500 users are grouped into a cell and the available bandwidth in each cell is assumed to be the same. Due to normalization, a user on average has throughput of 1, which means a cell has 500 units of throughput on average. Figure 7 shows how the expected number of complaints can be reduced by allocating a certain amount of resource to each cell. The resource amount is normalized by 500, which is the existing throughput consumed by a cell on average. The Water Filling algorithm (and thus the meta heuristic) achieves the same performance as the optimal algorithm. This is because, all of the users have very low probabilities to complain, i.e. $C_i > 0$ for $i = 1, 2, ..., M$ in Eq. (9), which makes the optimization problem **(P-2)** to be convex. This figure shows the optimal algorithm can reduce the expected number of complaints by 30% when 100% of additional resource is available. In comparison, to achieve the same goal, the baseline algorithm needs 240% additional resource. When the normalized resource is less than 10%, the Sweep algorithm reduces more than four times positives compared with the baseline algorithm.

The summary of reduced number of positives in the 3 experiments are given in Table 3. In experiment 2 and 3, the resource is normalized by the existing bandwidth in the video system/a cell. The table shows given insufficient amount of resource, the Sweep algorithm allocates the resources much more efficiently compared with the baseline.

Table 3: Reduced number of positives in the 3 experiments.

Experiment 1. Resource	1	10	100	398	1000	2510	3980	10000
The Sweep Algorithm	0.0775	0.759	6.755	21.0	42.2	81.3	99.2	100
Baseline	0.00842	0.0849	0.913	4.45	14.7	57.8	89.4	95.4
Experiment 2. Normalized Resource	0.000737	0.00293	0.0117	0.0465	0.185	0.737	1.17	2.93
The Sweep Algorithm	0.495	1.95	7.69	29.8	106	265	277	278
Baseline	0.245	0.980	3.98	17.1	84.5	237	240	241
Experiment 3. Normalized Resource	0.002	0.0112	0.0632	0.356	1.12	2	20	200
The Sweep Algorithm	2.32	12.1	49.0	141	246	314	611	654
Baseline	0.231	1.30	7.25	39.6	116	190	562	573

6. Discussion

There are several limitations of our work. First, the logistic regression model, while widely used, could be too simplified as it cannot depict a complex structure if there is. Second, we try to modify the label of an instance by allocating resource to alter its features, which is only effective to a subset of problems. Third, the current work assumes resource allocation modifies feature values linearly. While it works for a set of application scenarios, we hope to generalize the model to incorporate more sophisticated scenarios, for example, nonlinear or correlated models between resource allocation and feature values.

In the work, we target to reduce the total number of positives, which is a key performance indicator of service providers. However, this objective doesn't guarantee fairness among users. For example, with limited amount of resource, resource allocation gives higher priority to the users with medium service quality, and cannot improve service for users with the worst quality. Our future work will consider more complex model involving fairness.

7. Conclusion

This work studies how learning a classification model and allocating resource can be integrated coherently to maximize the expected number of satisfied users efficiently. This framework first trains a classifier based on the training data, and then applies this classifier as the objective function to guide resource allocation on the test data. Based on logistic regression, an optimal resource allocation algorithm and algorithms with low complexity are designed. Finally, the algorithms are applied to reduce unsatisfied viewers in a video transmission system and customer complaints in a cellular network. Compared with the baseline, the optimal algorithm we proposed can reduce customer complaints by more than four times. Future work includes extending the framework of learning and resource allocation to more classification models and more general resource allocation models.

Acknowledgments

This work was partially supported by NSF through grants CNS-1457060, CNS-1464421 and CCF-1423542.

References

E Baik, A Pande, C Stover, and P Mohapatra. Video acuity assessment in mobile devices. In *The 32nd IEEE International Conference on Computer Communications*, 2015.

Athula Balachandran, Vyas Sekar, Aditya Akella, Srinivasan Seshan, Ion Stoica, and Hui Zhang. Developing a predictive model of quality of experience for internet video. In *ACM SIGCOMM Computer Communication Review*, volume 43, pages 339–350. ACM, 2013.

Mohsen Bayati, Mark Braverman, Michael Gillam, Karen M Mack, George Ruiz, Mark S Smith, and Eric Horvitz. Data-driven decisions for reducing readmissions for heart failure: General methodology and case study. *PloS one*, 9(10):e109264, 2014.

Josep Ll Berral, Íñigo Goiri, Ramón Nou, Ferran Julià, Jordi Guitart, Ricard Gavaldà, and Jordi Torres. Towards energy-aware scheduling in data centers using machine learning. In *Proceedings of the 1st International Conference on energy-Efficient Computing and Networking*, pages 215–224. ACM, 2010.

Rajkumar Buyya, Rajiv Ranjan, and Rodrigo N Calheiros. Intercloud: Utility-oriented federation of cloud computing environments for scaling of application services. In *Algorithms and architectures for parallel processing*, pages 13–31. Springer, 2010.

An Jack Chan, Amit Pande, Eilwoo Baik, and Prasant Mohapatra. Temporal quality assessment for mobile videos. In *Proceedings of the 18th MobiCom*, pages 221–232. ACM, 2012.

Anshul Gandhi, Varun Gupta, Mor Harchol-Balter, and Michael A Kozuch. Optimality analysis of energy-performance trade-off for server farm management. *Performance Evaluation*, 67(11): 1155–1171, 2010.

Frank E Harrell. *Regression modeling strategies.* Springer Science & Business Media, 2001.

E Horvitz and T Mitchell. From data to knowledge to action: a global enabler for the 21st century. computing community consortium, v. 11. september 11, 2010.

David W Hosmer Jr and Stanley Lemeshow. *Applied logistic regression.* John Wiley & Sons, 2004.

Fei Huang, Zhipeng Jiang, Sanguo Zhang, and Suixiang Gao. Reliability evaluation of wireless sensor networks using logistic regression. In *Communications and Mobile Computing (CMC), 2010 International Conference on*, volume 3, pages 334–338. IEEE, 2010.

S Shunmuga Krishnan and Ramesh K Sitaraman. Video stream quality impacts viewer behavior: inferring causality using quasi-experimental designs. *Networking, IEEE/ACM Transactions on*, 21(6):2001–2014, 2013.

Gunho Lee, Niraj Tolia, Parthasarathy Ranganathan, and Randy H Katz. Topology-aware resource allocation for data-intensive workloads. In *Proceedings of the first ACM asia-pacific workshop on Workshop on systems*, pages 1–6. ACM, 2010.

Ricky KP Mok, Edmond WW Chan, and Rocky KC Chang. Measuring the quality of experience of http video streaming. In *Integrated Network Management (IM), 2011 IFIP/IEEE International Symposium on*, pages 485–492. IEEE, 2011.

P. Paudyal, F. Battisti, and M. Carli. A study on the effects of quality of service parameters on perceived video quality. In *Procs. of 5th EUVIP 2014*, December 2014.

Meng Qu, Hengshu Zhu, Junming Liu, Guannan Liu, and Hui Xiong. A cost-effective recommender system for taxi drivers. In *Proceedings of the 20th ACM SIGKDD*, pages 45–54. ACM, 2014.

Zukang Shen, Jeffrey G Andrews, and Brian L Evans. Adaptive resource allocation in multiuser ofdm systems with proportional rate constraints. *Wireless Communications, IEEE Transactions on*, 4(6):2726–2737, 2005.

Guocong Song and Ye Li. Cross-layer optimization for ofdm wireless networks-part i: theoretical framework. *Wireless Communications, IEEE Transactions on*, 4(2):614–624, 2005.

Theja Tulabandhula and Cynthia Rudin. Machine learning with operational costs. *The Journal of Machine Learning Research*, 14(1):1989–2028, 2013.

Theja Tulabandhula and Cynthia Rudin. On combining machine learning with decision making. *Machine Learning*, 97(1-2):33–64, 2014.

Appendix A. Proof of the Optimality

In order to prove Proposition 2, KKT conditions are derived to capture the characteristics of this optimization problem first, and Lemma 3 and Lemma 4 finish the proof. Lemma 3 shows for the instances with resource allocation, there is going to be a water level. Lemma 4 argues that one optimal solution allocates resource only to a contiguous subset of C_i^s-s. The Sweep algorithm tests all contiguous subset of C_i^s-s and allocates resource with water filling. Therefore, it can find one of the optimal solutions definitely.

The Lagrangian of **(P-2)** is

$$L(\boldsymbol{s}, \boldsymbol{\lambda}, v) = \sum_{i=1}^{M} \frac{1}{1+\exp(C_i+s_i)} - \sum_{i=1}^{M} \lambda_i s_i - v(S - \sum_{i=1}^{M} s_i) = \sum_{i=1}^{M} \frac{1}{1+\exp(C_i+s_i)} - Sv + \sum_{i=1}^{M} (v-\lambda_i) s_i \quad (15)$$

Since the constraints are linear and the objective function is smooth, the optimal solutions have to meet the KKT conditions as follows:

$$-\frac{\exp(C_i+s_i)}{(1+\exp(C_i+s_i))^2} + v - \lambda_i = 0, \quad i = 1, 2, ..., M; \quad (16)$$

$$\sum_{i=1}^{M} s_i = S; \quad (17)$$

$$\lambda_i s_i = 0, \quad i = 1, 2, ..., M; \quad (18)$$

$$v \geq 0; \quad (19)$$

$$\lambda_i \geq 0 \text{ for } i = 1, 2, ..., M. \quad (20)$$

Constraints (18) and (16) show when $\lambda_i = 0$, there is $\frac{\exp(C_i+s_i)}{(1+\exp(C_i+s_i))^2} = v$; otherwise $s_i = 0$.

Lemma 3 *There is one optimal solution that satisfies either 1) $s_i = 0$, or 2) $C_i + s_i$ is a constant independent of i.*

Proof Denote $\frac{1}{1+\exp(x)}$ by $\eta(x)$, and its derivative is

$$\eta'(x) = \frac{-\exp(x)}{(1+\exp(x))^2} < 0. \quad (21)$$

As shown by Fig. 8, $\eta'(x)$ is symmetric on the vertical line $x = 0$. $\eta(x)$ is concave when $x \leq 0$ and convex when $x \geq 0$. When s units of resource is allocated to point 1, it can be moved to point 3. The area above the curve staring from point 1 and ending at point 3 corresponds to the reduction of the objective function.

Denote the set of indices with resource allocation by I_a, i.e. $s_i > 0$ for $i \in I_a$. We have $\eta'(C_i + s_i) = -v$, i.e., $C_i + s_i = \pm \log(-1 + \frac{1}{2v} + \frac{\sqrt{1-4v}}{2v})$ for $i \in I_a$. Next, we are going to prove $s_i + C_i = \log(-1 + \frac{1}{2v} + \frac{\sqrt{1-4v}}{2v})$. If $v = \frac{1}{4}$, $C_i + s_i = 0$ for $i \in I_a$, the proof is done. Otherwise, assume $C_i + s_i < 0$ for $i \in I_l$, and $C_i + s_i > 0$ for $i \in I_r$, $I_l \cup I_u = I_a$, $I_l \cap I_u = \emptyset$.

We first prove there is at most one point in I_l. Assume there are at least two points in I_l, and their indices are $i_{l,1}$ and $i_{l,2}$. With a small value Δ, assign $s'_{i_{l,1}} = s_{i_{l,1}} + \Delta$ and $s'_{i_{l,2}} = s_{i_{l,2}} - \Delta$. Then, $s'_{i_{l,1}}$ and $s'_{i_{l,2}}$ will achieve a decrease in the objective function, because $\eta(x)$ is concave when $x < 0$. Therefore, the left set I_l includes at most one point.

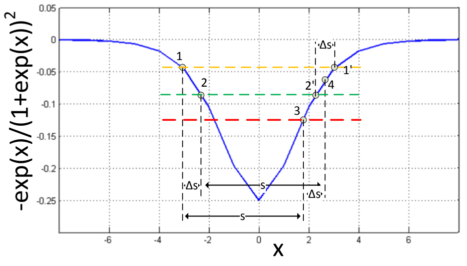


Figure 8: An example showing the resource allocation.

Denote the index of the only point in I_l by i_l, we are going to prove when the resource s_{i_l} allocated to i_l is given to the right set I_r, the objective function cannot increase. For any point in the I_r, denote its index by i_r. Since $C_{i_l} + s_{i_l} = -(C_{i_r} + s_{i_r})$, $\eta'(C_{i_l} + s_{i_l} - \beta) = \eta'(C_{i_r} + s_{i_r} + \beta)$ for $0 \leq \beta \leq s_{i_l}$. Moving the resource s_{i_l} from i_l and giving it to i_r will keep the objective function unchanged. Moreover, it is easy to prove that when s_{i_l} is given to more than one points in I_r, the objective function will decrease. Therefore, there is one optimal solution where the left set I_l is empty and $\forall i \in I_a$, $s_i + C_i = \log(-1 + \frac{1}{2v} + \frac{\sqrt{1-4v}}{2v})$. ■

Lemma 4 *There is one optimal solution, whose indices in C_i^s-s with $s_i > 0$ are contiguous. Here C_i^s-s is the list obtained by sorting C_i-s in ascending order.*

Proof Assume for one of the optimal solutions that meets Lemma 3, i-s are not contiguous. There are a left subset, a right subset and at least one point in the middle. Only the left subset and right subset are allocated resource and C^* is the water level after resource allocation, i.e., $C_i + s_i = C^* = \log(-1 + \frac{1}{2v} + \frac{\sqrt{1-4v}}{2v}) > 0$ for i in the left subset and the right subset. Denote the index of a point in the left subset by I_l, the index of a point in the middle by I_m, and the index of a point in the right subset by I_r. Resource of $C^* - C_{I_l}$ is consumed to move point I_l to the level C^*. The same amount of resource can move I_m to $C_{I_m} + C^* - C_{I_l}$.

Since $\lambda_{I_m} > 0$, according to (16) we have $\eta'(C_{I_m}) = \eta'(-C_{I_m}) > -v = \eta'(C^*)$. If $C_{I_m} \geq 0$, we have $C_{I_m} > C^*$, since $\eta'(x)$ is an increasing function when $x \geq 0$. However, this contradicts the assumption that $C_{I_m} < C_{I_r} + s_{I_r} = C^*$. Therefore, we have $C_{I_m} < 0$ and $-C_{I_m} > C^*$.

Since $C_{I_l} < C_{I_m} < 0$, $C^* > 0$, $C_{I_m} + C^* < 0$ and $\eta(x+e) - \eta(x)$ is an increasing function of x when $x \geq 0$, for any $e > 0$, we have $\eta(C^* + (C_{I_m} - C_{I_l})) - \eta(C^*) < \eta(-C_{I_l}) - \eta(-C_{I_m})$. Since $\eta(x_1) - \eta(x_2) = \eta(-x_2) - \eta(-x_1)$, then we have $\eta(C_{I_m} + (C^* - C_{I_l})) - \eta(C^*) < \eta(C_{I_m}) - \eta(C_{I_l})$, i.e., $\eta(C_{I_m} + (C^* - C_{I_l})) + \eta(C_{I_l}) < \eta(C^*) + \eta(C_{I_m})$. This means allocating $C^* - C_{I_l}$ units of resource to I_m leads to a decrease of the objective function, which contradicts with the assumption that the original solution is one of the optimal solutions. ■

Fig. 8 gives an example for this proof. Point 3 is the water level C^*. Point 1 and point 2 are I_l and I_m, respectively. To move the point 1 to the point 3, s units of resource are needed. The same amount of resource can move point 2 to point 4, and meanwhile $s - \Delta s$ units of resource are consumed by moving point 2 to point 3. When Δs units of resource are used to move 3 to 4, it has more reduction on the objective function compared with using Δs to move point $2'$ to point $1'$. Here point $1'$ and point 1 are symmetric to y-axis and the same goes for point $2'$ and point 2. Therefore, allocating s units of resource to point 2 is better than allocating them to point 1.

Since resource has to be allocated to a contiguous indices on C_i^s-s, the Sweep algorithm tests all possibilities and can find one of the optimal solutions definitely.