Website Optimization Problem and Its Solutions

Shuhei Iitsuka
The University of Tokyo
Hongo 7-3-1, Bunkyo-ku
Tokyo, Japan
iitsuka@weblab.t.u-tokyo.ac.jp

Yutaka Matsuo
The University of Tokyo
Hongo 7-3-1, Bunkyo-ku
Tokyo, Japan
matsuo@weblab.t.u-tokyo.ac.jp

ABSTRACT

Online controlled experiments are widely used to improve the performance of websites by comparison of user behavior related to different variations of the given website. Although such experiments might have an important effect on the key metrics to maximize, small-scale websites have difficulty applying this methodology because they have few users. Furthermore, the candidate variations increase exponentially with the number of elements that must be optimized. A testing method that finds a high-performing variation with a few samples must be devised to address these problems.

As described herein, we formalize this problem as a website optimization problem and provide a basis to apply existing search algorithms to this problem. We further organize existing testing methods and extract devices to make the experiments more effective. By combining organized algorithms and devices, we propose a rapid testing method that detects high-performing variations with few users. We evaluated our proposed method using simulation experiments. Results show that it outperforms existing methods at any website scale. Moreover, we implemented our proposed method as an optimizer program and used it on an actual small-scale website. Results show that our proposed method can achieve 57% higher performance variation than that of the generally used A/B testing method. Therefore, our proposed method can optimize a website with fewer samples. The website optimization problem has broad application possibilities that are applicable not only to websites but also to manufactured goods.

Categories and Subject Descriptors

G.3 [**Probability and Statistics**]: Experimental design

Keywords

Website optimization; A/B testing; Controlled experiments

KDD'15, August 10-13, 2015, Sydney, NSW, Australia.
© 2015 ACM. ISBN 978-1-4503-3664-2/15/08 ...$15.00.
DOI: http://dx.doi.org/10.1145/2783258.2783351.

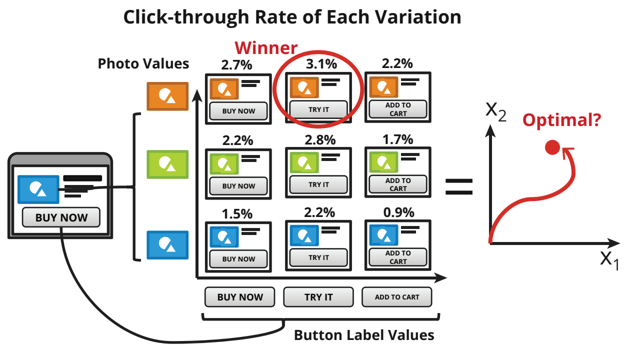

Figure 1: Overview of online controlled experiment.

1. INTRODUCTION

Online controlled experiments using experimental approaches are becoming popular among website owners as a method to make websites more profitable. We present an overview of this methodology in Figure 1. It compares user behavior related to different variations that are modified in the respect of design elements and functions from the original website. The goal is to find a high-performing variation by measuring the key metrics, which include the click-through rate (CTR), page views per session, and time on site. For example, e-commerce websites might improve their revenue by maximizing CTR of the purchase button on their websites. The main goal of online controlled experimentation is to ascertain the best label that enhances use of the click button. When we assume that the website consists of such elements, a website variation is expressed as a solution for the combination of element values. In other words, online controlled experimentation can be regarded as a search problem to ascertain an optimal solution. Online controlled experimentation is often designated as split testing, A/B testing, or randomized experimentation[19].

Small changes to a website are known to have a strong effect on profitability. For example, United States President Barack Obama's digital campaign team conducted a simple experiment on his official website to raise money for the presidential election of 2008[1]. Figure 2 shows the four button labels and six eye-catching media tested on the home page.

[1]How Obama Raised $60 million by Running a Simple Experiment `http://blog.optimizely.com/2010/11/29/how-obama-raised-60-million-by-running-a-simple-experiment/`

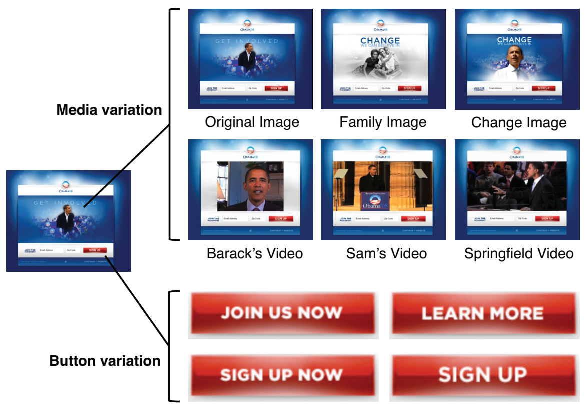

Figure 2: Variation of buttons and media used for President Obama's official website.

The team assigned visitors to each combination and compared their sign-up rates. Consequently, the variation with the picture of his family and "Learn More" label earned the highest sign up rate. They raised an additional $60M in donations by application of the best performing variation to their home page. Another experiment related to Microsoft's Bing search engine shows that a slight change of font colors led to a $10M annual revenue improvement[18]. Online controlled experiments are conducted using various large websites, yielding improvements in their business fundamentals.

However, small-scale websites have difficulty in applying such concepts because they have few users. Large numbers are crucially important to support the power of statistical inference[19]. It is possible to extend the experimental period to collect samples, but doing so might allow seasonal factors to affect the experiment results. For example, users might be attracted to designs with the color scheme of orange and black during the Halloween season, but they might prefer red and green during the Christmas season[25]. In addition, the number of candidate variations increases exponentially with numerous elements to optimize. Therefore, small-scale websites demand a method for rapid testing that can identify a high-performing variation, even for a site with few users.

Some studies have mathematically formalized this problem as a multi-armed bandit problem by assuming each variation uniformly as a solution. However, such methods use no structure of the solution to a full extent. By assuming each solution as a combination of elements, the problem is definable as a combinational optimization problem. Such a defined problem enables the application of heuristic algorithms to find an approximate solution quickly. Local search, a popular heuristic algorithm, is applicable even if the evaluation function is nonlinear, but it has the important shortcoming that it can fall into a local optimum depending on the initial solution[1]. Nevertheless, each element's effects can be evaluated by analyzing the samples collected from the entire solution space. To do so, it is necessary to incorporate an assumption that the performance of the given solution is expressed using a linear combination of them[12]. The methods have benefits and shortcomings, but we can integrate them for effective exploration, analyze each element's effect to find the good initial solution, and start finding better solutions by evaluating neighbor solutions.

As described in this paper, we formulate such problems as *website optimization problems*, which are instances of combinational optimization problems. We also organize existing testing methods as optimization algorithms and extract the devices for effective experiments. We propose a fast testing method combining these existing testing methods and devices. Whereas existing A/B testing repeats two-group comparison of variations, our proposed method estimates the best performing value of each element first, then it starts local search from their combination. Finally, we evaluate our proposed method using a simulation experiment with evaluation functions estimated from log data stored in an actual large-scale website. We also implemented our proposed method as an optimizer program and introduced it into an actual small-scale website. Results show that our proposed method can reach higher performing variation with the same number of users compared to the generally used A/B testing method, which means that it can optimize a website with fewer users.

The contributions of this paper are summarized as follows:

- We formulated the website optimization problem, which provides researchers and website owners the mathematical basis to work on this problem.
- We organized existing testing methods and devices, which enables researchers to capture them as algorithms for an optimization problem and enables website owners to produce new methods by combining them.
- We proposed a rapid testing method that can find high-performing variation with few users. Website owners can make a website more profitable quickly by implementing it.

The remainder of this paper is organized as explained below. We introduce related works in the next section. In Section 3, we explain existing testing methods that are useful for online controlled experiments. In Section 4, we formalize the website optimization problem and organize existing testing methods. In Section 5, we propose a rapid testing method. We evaluate our proposed method with experiments using actual websites in Section 6. After a discussion presented in Section 7, we conclude this paper in Section 8.

2. RELATED WORKS

Data mining technology is playing an important role in improving website performance. E-commerce websites have been studied to produce benefits from insight about user behavior extracted from their log data[17]. Internet ads are examined intensively. Chakrabarti et al. use word relevance between ads and web pages to predict ad CTR[10]. Li et al. introduce reinforcement learning technology to increase CTR by formalizing the problem as an instance of a *multi-armed bandit problem*, assuming each variation as an arm, each user as a trial, and the observed click as a reward[21]. This problem is generalized as a contextual-bandit problem assuming that each user has individual features[20]. User behavior stored in the log data is used for website content personalization by predicting user characteristics and demographic information[16]. Not only the log data stored in the website but also social data obtained by crawling social media are used to improve the website performance[8].

Widely diverse IT companies are introducing online controlled experimentation, which constitutes an online approach that modifies the website contents based on real-time implicit user feedback. They are improving the experimentation process by inventing their own devices to produce benefits. Google parameterizes their search results pages and groups parameters into subsets, which enables optimization of both visible and invisible elements simultaneously[24]. Facebook is improving their experiment platform with a framework to estimate the dependence effect of user-item experiments[6]. They also implement their own programming language specialized for online controlled experimentation[7]. Practical tips and avoidable pitfalls that are specific to the web are examined through experiments conducted using actual websites at Microsoft[13]. Yahoo! proposes a framework to optimize CTR and post-click engagement of the articles[3].

Other studies have been conducted to increase the experimentation rapidity and effectiveness. Deng et al. propose a method to reduce the variability of the metrics using historical data collected before the experiment[14]. Historical data are also used to increase the speed of content recommendation[2]. Meta-heuristic algorithms such as genetic algorithms are also applied to optimize the website contents[4]. The design of the key metrics to optimize is also important. Sculley et al. propose a bounce rate as an important metric to estimate the performance of sponsored search ads[22].

Although some studies are being conducted through online controlled experimentation on websites by application of mathematical models, few studies specifically examine small-scale websites that serve hundreds of users per day. As described herein, we propose an approach to formalize this problem as a combinational optimization problem. Moreover, we provide a basis to generate new optimization methods that are applicable irrespective of the website scale.

3. TESTING METHODS FOR ONLINE CONTROLLED EXPERIMENT

In this section, we introduce popular testing methods used presently for online controlled experiments. The most simplified form of controlled experiment is *A/B testing*. It compares the performance of variation A (control), which is currently used, and variation B (treatment), which is modified. Fundamentally, the modification is applied to a few elements to assess and quantify the effects of the change. Users are distributed randomly to one of the two variations; then their behaviors are observed. Through comparison of the collected observations, one can ascertain whether the modification has good effects on the website, or not. One can make a reliable decision by introducing statistical hypothesis testing of the collected samples, such as Student's t-test.

Multivariate testing (MVT) modifies multiple elements simultaneously, whereas A/B testing often denotes simpler experiments. It generates all possible variations by combining the value of elements and selects the candidate variations from them. Actually, MVT assigns users to candidate variations and compares user behavior[19]. The analysis of the experiment result becomes more complicated than A/B testing, but it enables optimization of multiple elements at the same time[19]. Two implementations of MVT are *full factorial design* and *fractional factorial design*.

Full factorial design tests all possible combinations. It finds the best-performing variation while strictly considering the interactive effects of elements, but it entails the important shortcoming that it requires numerous samples because the number of candidate variations increases exponentially with the number of elements to optimize. Statistical hypothesis testing for two groups cannot be applied because it tests over two variations simultaneously. Multiple comparison adjustment can be used to compare the variations, but it is not feasible when the variations become numerous. Actually, MVT enables identification of the optimal variation from the candidates considering interactive effects, but makes it difficult to collect sufficient samples and to make decisions based on statistical testing.

However, fractional factorial design tests the subset of the possible combinations with the assumption that high-order interactions are negligible, which is called the sparsity-of-effects principle. The subset is designed a priori using an *orthogonal array* to choose and to estimate the main and interactive effects of the elements of interest. The orthogonal array must be sufficiently large to allocate the elements, the main effects of which we examine. If the estimation of some interactive effects of two elements is desired, then it is also possible to include them into the allocation. The main effects and low-level interactive effects are calculated using *analysis of variance* (ANOVA), which supports inference of the optimal solution by combining the best performing value of each element considering non-negligible interactions[12].

This idea of fractional factorial design was proposed originally in *Design of Experiments*, for agricultural and manufacturing fields. It has been used to measure each element's effect with small samples. It is also applied to identify a well-performing parameter value set for meta-heuristic algorithms[1]. Fractional factorial design is used in many areas to reduce the number of variations to be tested and to make the experiment effective. This effectiveness derives from the assumption that the performance is expressed as the linear combination of the main effects and low-level interactive effects of the elements.

The *Bandit Algorithm* is also used to optimize a website by characterizing the problem as a multi-armed bandit problem[25]. It provides an optimization method to find the optimal solution in the balance between *exploitation* and *exploration*. Exploitation takes the solution with the maximum expected value, whereas exploration takes the solution for which the performance is unobserved or uncertain. A high-performing solution can be found faster by avoiding wasting samples for well-known inferior solutions and by allocating samples to promising solutions. To maximize the sum of the rewards, some algorithms are proposed, such as ϵ-greedy, Softmax, and UCB[5].

4. WEBSITE OPTIMIZATION PROBLEM

In this section, we propose the formalization of an online controlled experiment as a combinational optimization problem that enables the application of widely diverse search algorithms. Then we select *local search* as an example of a heuristic algorithm and explain how it can solve the problem. Finally, we propose the organization of existing testing methods, assuming them as solutions for a website optimization problem.

4.1 Problem Definition

We explain the proposed formalization of online controlled experiment here. Website optimization is a process to seek the best performing variation of the given website. The candidate variations are modified in some aspects of the design elements and features. The performance is then evaluated using the key metrics, which include CTR, page views per session, time on site, and others. Each variation consists of elements such as buttons and labels on the web page. Therefore, it can be expressed as a combination of elements. In addition, this problem must be resolved under an environment in which feedback is given incrementally as the variation is shown to users.

Consider a variation $\boldsymbol{x} = (x_1, \cdots, x_m)$ expressed as a combination of m elements that form the website. The value of each element x_i is discrete and selected from the set of values $V_i = \{v_{i1}, \cdots, v_{il_i}\}$, assuming that l_i is the number of values of element x_i. For example, consider that a variation $\boldsymbol{x}$ of an e-commerce website is expressed as a combination of three elements: the product photograph x_1, copy text x_2, and the color of the call-to-action button x_3. In this case, the color x_3 is selected from the set of the color values $V_3 = \{\text{red}, \text{blue}, \text{green}, \text{yellow}\}$. We specifically examine the optimization of a single web page and assume that the elements are extracted from the web page arbitrarily here.

The goal of website optimization problem is to find the optimal solution $\boldsymbol{x}^*$ which maximizes the key metrics expressed by the evaluation function $f(\boldsymbol{x})$. However, we cannot observe the evaluation value $f(\boldsymbol{x})$ directly. All we can observe is user behavior feedback expressed as an observed value y derived from a probability distribution $p(y|\boldsymbol{x})$ when the variation $\boldsymbol{x}$ is displayed to a user. The evaluation value $f(\boldsymbol{x})$ can be estimated using a conditional expectation value $\mathbb{E}[y|\boldsymbol{x}]$. Therefore, we propose formalization of the website optimization problem as follows.

Website Optimization Problem

Find the solution $\boldsymbol{x}^*$ which satisfies the following equation.

$$\boldsymbol{x}^* = \arg\max_{\boldsymbol{x} \in X} \mathbb{E}[y|\boldsymbol{x}] \ s.t. \ y \sim p(y|\boldsymbol{x})$$

The website optimization problem is a combinational optimization problem to ascertain the optimal solution $\boldsymbol{x}^*$, which maximizes the conditional expectation value of the key metrics $\mathbb{E}[y|\boldsymbol{x}]$ from the solution space X. An observed value y is derived from the probability distribution $p(y|\boldsymbol{x})$.

4.2 Local Search for Website Optimization Problem

We formalized the website optimization problem as a combinational optimization problem. Therefore, widely diverse search algorithms can be introduced into it. Especially, a heuristic algorithm that finds an approximate solution in a short amount of time can be a promising solution because it is necessary in most cases to optimize a website that has few users. We explain how we can introduce heuristic algorithms to this problem by choosing an example of local search that is applied widely to optimization problems.

We present the local search algorithm in Algorithm 1. Local search seeks better solutions by making small changes on the current solution standing on the assumption called the proximate optimality principle, which assumes that good solutions have mutually similar structures[26]. Local search starts exploring the optimal solution from the initial solution $\boldsymbol{x} \in X$ selected from the candidates. It generates neighbor solutions X' by modifying one element of the current solution $\boldsymbol{x}$. The current solution moves to the neighbor solution if any neighbor solution exists with an evaluation value exceeding the current solution's value. The set of these two operations *neighbor solution generation* and *solution move* is iterated until no improvement is found or until a time limit is reached.

Algorithm 1 Local search method.

```
Choose $\boldsymbol{x} \in X$.
repeat
    $X' \leftarrow Neighbors(\boldsymbol{x})$    ▷ Neighbor solution generation
    $\boldsymbol{x} \leftarrow Move(\boldsymbol{x}, X')$    ▷ Solution move
until No improvement is made or a time limit is reached.
return $\boldsymbol{x}$
```

In the context of an online controlled experiment, the neighbor solution generation corresponds to changing of a single element x_i of the current website $\boldsymbol{x}$ by swapping the value selected from the value set V_i. The current solution moves if there is any neighbor solution with a higher evaluation value estimated by the conditional expected value $\mathbb{E}[y|\boldsymbol{x}]$.

4.3 Organization of Existing Testing Methods

Table 1: Organization of existing testing methods.

Testing method	Search algorithm	Device
A/B testing	Local search	None
Full factorial MVT	Brute-force	None
Fractional factorial MVT	Brute-force	Linear assumption
Bandit algorithm	Brute-force	Flexible allocation

We propose the organization of existing testing methods described in Section 3 based on our proposed website optimization problem definition. The organization is presented in Table 1 in the form of search algorithms and devices for efficient experiments. Repeating A/B testing on multiple elements corresponds to employing local search, which starts from the current solution. It generates neighbor solutions X' by changing the value of a given element x_i, which consists of the current solution $\boldsymbol{x}$. One experiment is completed by evaluating the neighbor solutions and by moving the current solution to a better one. One can repeat A/B testing by changing the element to optimize in the subsequent experiment.

Full factorial MVT evaluates all possible combinations, which corresponds to a brute-force search. Fractional factorial MVT testing also uses brute-force search, but it evaluates only the subset of the possible combinations assuming that the key metrics are expressed as a linear combination of each element's effect. We can extract this assumption as a device to reduce the number of solutions to test. We designate this device herein as a *Linear assumption.*

The bandit algorithm also uses brute-force search, but the difference is that it controls the sample size allocation to each solution $\boldsymbol{x}$ according to the expected value $\mathbb{E}[y|\boldsymbol{x}]$. Assigning more samples to more promising solutions streamlines and hastens the exploration. We designate this idea as *Flexible allocation* in this paper.

To implement flexible allocation, an alternative method named *Racing algorithm* exists: it has been applied to find better solutions effectively in the field of reinforcement learning and meta-heuristics configuration problems[9, 15]. Although the bandit algorithm requires parameter-tuning to maximize its performance, the racing algorithm is simple; it requires no parameter tuning. We introduce the racing algorithm here, which is preferred under circumstances in which the website characteristics are unknown.

The racing algorithm calculates the confidence interval $[\mu_{L\boldsymbol{x}}, \mu_{U\boldsymbol{x}}]$ of each solution's evaluation value $f(\boldsymbol{x})$ every time it obtains a new observed value y. We designate $\mu_{L\boldsymbol{x}}$ *lower confidence limit* and $\mu_{U\boldsymbol{x}}$ *upper confidence limit* here. When the solution $\boldsymbol{x}_{\text{loser}}$ has an upper confidence limit $\mu_{U\boldsymbol{x}_{\text{loser}}}$ that is inferior to all other solutions' lower confidence limits, the solution is removed from the candidate solutions as the loser solution. In other words, when the solution $\boldsymbol{x}_{\text{loser}}$ satisfies $\mu_{U\boldsymbol{x}_{\text{loser}}} < \min_{\boldsymbol{x}\in X\setminus\{\boldsymbol{x}_{\text{loser}}\}}(\mu_{L\boldsymbol{x}})$, it is dropped from candidates X. If the number of candidate solutions becomes one, then the left solution is taken immediately as the optimal solution. In other words, when the solution $\boldsymbol{x}_{\text{winner}}$ satisfies $\mu_{L\boldsymbol{x}_{\text{winner}}} > \max_{\boldsymbol{x}\in X\setminus\{\boldsymbol{x}_{\text{winner}}\}}(\mu_{U\boldsymbol{x}})$, it is adopted as the optimal solution. The observation is completed promptly.

5. PROPOSED FAST TESTING METHOD

We propose a rapid testing method that can find high-performing variation with few users by combining the devices we extracted in Section 4. This method explores the optimum solution effectively. It is applicable irrespective of the website scale.

The inspiration for this testing method is the following. All candidates can be evaluated with sufficient samples if the given website is large and if it has many visitors. However, small-scale websites cannot evaluate all of them because the users are few. It takes a long time to gather sufficient samples. Therefore, it is necessary to estimate the optimal solution with few samples, even assuming that the evaluation function is described as a linear combination of each element's effect. Local search is useful to find a better solution that is ignored by the assumption above if some samples remain after the estimation. In addition, the exploration can be accelerated by the application of flexible allocation implemented with the racing algorithm. A similar method is proposed by Coys et al. The method finds a good parameter value set by conducting fractional factorial design experiments before introducing local search to find a better value set for meta-heuristic algorithms[12]. We can use a more generic assumption such as Gaussian process, but we use the linear assumption here to simplify the testing method.

We present the outline of our proposed method, named *Linear Assumption and Local Search* (LALS) in Figure 3 and the algorithm in Algorithm 2. The required constants for LALS are the significance level α, the statistical power

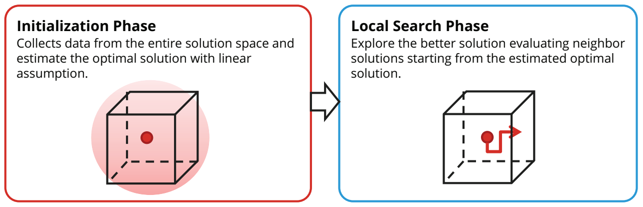
Initialization Phase
Collects data from the entire solution space and estimate the optimal solution with linear assumption.
Local Search Phase
Explore the better solution evaluating neighbor solutions starting from the estimated optimal solution.

Figure 3: Outline of the proposed method.

$1-\beta$, the effect size to detect Δ, the sample size bound N and the set of candidate solutions X. The significance level α stands for the probability of making a Type I error, whereas β represents the probability of making a Type II error. The convention values are proposed as $\alpha = 0.05$ and $1-\beta = 0.8$ in an earlier study[11]. We use them in the following experiments described in Section 6. The effect size Δ is the mean difference between two groups standardized by the standard deviation. This value is designated according to how much improvement the website owner anticipates from the results of the experiment.

Algorithm 2 Algorithm of LALS.

```
Require: $\alpha$ as the significance level.
Require: $1-\beta$ as the power.
Require: $\Delta$ as the effect size.
Require: $N$ as the sample size bound.
Require: $X$ as the set of solutions.
  Set $Y$ as an empty set for observed data.
  Set $n \leftarrow 0$ as the total number of observations.
  Set $N_1 \leftarrow N_{ANOVA}(\alpha, \beta, \Delta, X)$
  Set $N_2 \leftarrow N_{T-TEST}(\alpha, \beta, \Delta)$
  $\boldsymbol{x}^*, Y, n \leftarrow Initialization(X, Y, N_1, n)$
  while $n < N$ do
      $\boldsymbol{x}^*, Y, n \leftarrow Move(\boldsymbol{x}^*, X, Y, N_2, n)$
  return $\boldsymbol{x}^*$ as the optimal solution.
```

Estimation of the optimal solution with a linear assumption device requires sample size N_1, which is sufficiently large to detect the difference of the evaluation value across the element's value. This time, we use the sample size calculated for the one-way ANOVA N_{ANOVA}. Observation values are needed for each value of the elements. Therefore, we assume the number of groups k for the one-way ANOVA by the sum of the number of values as $k = \sum_{i=1}^{m} l_i$. We use the sample size required for Student's t-test $N_{\text{T-TEST}}$ as the sample size N_2 to evaluate neighbor solutions in the local search. The calculation of N_1 and N_2 follows the method proposed in an earlier study[11].

Actually, LALS consists of two search phases: *initialization* and *local search*. The initialization phase estimates the optimal solution from the collected data using a linear assumption device. It collects data by presenting a solution that was selected randomly from the candidates X for N_1 times. The estimated optimal solution is used as the initial solution for the following local search phase.

In the local search phase, it generates neighbor solutions X' by modifying the value of randomly selected element x_i of the current solution $\boldsymbol{x}$. We choose one neighbor solution $\boldsymbol{x}'$ randomly from neighbor solutions X' for comparison with the current optimal solution. Sample size N_2 is allocated to the neighbor solution $\boldsymbol{x}'$ to collect the observed values. After the observation is completed, expectation $\mathbb{E}[y|\boldsymbol{x}]$ of each solution is calculated. If the expectation of the neighbor solution is higher than the current one, then the optimal solution is updated with the neighbor solution. The local search phase finds the optimal solution, which cannot be found in the initialization phase because of non-negligible interactive effects of the elements. The local search phase lasts until the number of the observed data n reaches the sample size bound N.

Additionally, the search process can be accelerated by the application of a flexible allocation device implemented with the racing algorithm. Applying the racing algorithm to the initialization phase is expected to fix the initial solution before sample size N_1 is achieved. The solution can also be moved as soon as a markedly superior neighbor solution is found before the sample size N_2 is reached in the local search phase. Each phase can be made faster by avoiding wasting samples for markedly inferior solutions. We propose a testing method *LALS+*, which introduces flexible allocation to LALS.

Table 2: Testing methods used for comparison in the experiment.

Method	*Initialization*	*Move*
BF	Random	N/A
LA	Linear assumption	N/A
LS	Random	Local search
LALS	Linear assumption	Local search
LALS+	Linear assumption + Racing algorithm	Local search + Racing algorithm

Table 3: Problem settings to be simulated.

Problem	Evaluation Function	Sample Size Bound N
1	Linear $f_1(\boldsymbol{x})$	500 ($N < N_1$)
2	Linear $f_1(\boldsymbol{x})$	2000 ($N > N_1$)
3	Nonlinear $f_2(\boldsymbol{x})$	500 ($N < N_1$)
4	Nonlinear $f_2(\boldsymbol{x})$	2000 ($N > N_1$)

6. EXPERIMENT

In this section, we evaluate our proposed methods using experiments. We first evaluate our proposed methods using the simulation experiments. Then we evaluate our proposed method irrespective of the scale of the website by applying it to both real large-scale and small-scale websites.

6.1 Testing Methods

The testing methods used for comparison in the evaluation experiments are presented in Table 2. Brute Force (BF) randomly selects a solution to observe and collect observed values until it reaches the termination criterion. This method corresponds to full factorial MVT and returns the optimal solution: that with the highest expected value. Linear Assumption (LA) also selects a solution randomly to observe and collect the observed values as BF does, but it returns the optimal solution combining the best performing value of each element. In other words, this method corresponds to a kind of fractional factorial MVT which does not consider any interactive effect. This method is designed to show the upper limit performance of the ideal case in which the evaluation function is perfectly linear, which rarely occurs with actual websites. Local Search (LS) selects the initial solution randomly, consuming no sample, and starts the local search. This method corresponds to simple A/B testing, which starts from one possible solution. The sample size N_2 is allocated to each neighbor solution for evaluation. We use the method presented above as a baseline for comparison with our proposed ones. One proposed method, LALS, collects N_1 observed samples by choosing a solution randomly from the candidates. After the data collection, it estimates the optimal solution with a linear assumption device and uses the estimated optimal solution as the initial solution of the following local search phase. LALS+ introduces flexible allocation implemented with the racing algorithm to LALS.

6.2 Simulation Experiment on an Artificial Problem

We set up artificial problems assuming various scales of the websites and key metrics. Then we evaluated our proposed methods. Results show that our proposed methods can reach high-performing solutions with fewer samples than when using baseline methods.

In this experiment, we assume various situations and set the artificial problems shown in Table 3. We prepare two evaluation functions: a linear function $f_1(\boldsymbol{x})$ and a nonlinear function $f_2(\boldsymbol{x})$. We defined the evaluation functions as shown below.

$$\begin{aligned} f_1(\boldsymbol{x}) &= x_1 + x_2 + x_3 - x_4 - x_5 - x_6 + N(0,1) \\ f_2(\boldsymbol{x}) &= x_1 + x_2 + x_3 - x_4 - x_5 - x_6 - x_1x_2 + N(0,1) \end{aligned}$$

For the number of variables $m = 6$, each variable has three candidate values: $V_i = \{0, 1, 2\}$. In the equations, $N(\mu, \sigma)$ denotes a noise effect derived from a normal distribution in which μ is the mean and σ is the standard deviation. We set the linear function $f_1(\boldsymbol{x})$ to make both a plus sign and a minus sign appear an equal number of times. Therefore, either $x_i = 0$ or $x_i = 2$ is the optimal value for each variable. The nonlinear function $f_2(\boldsymbol{x})$ includes a member $-x_1x_2$, which expresses negative interaction of elements x_1 and x_2, which prevents $(x_1, x_2) = (2, 2)$ from being the optimal value set on the contrary to $f_1(\boldsymbol{x})$. We also separate the problems according to whether the sample size bound N is larger than the sample size N_1, or not. When N is smaller than N_1, the testing method cannot progress to the local search phase, but it can explore better solutions with local search after the initialization phase if N is larger than N_1. We set constants as $\alpha = 0.05, 1 - \beta = 0.8$, and $\Delta = 0.2$. Sample sizes are calculated as $N_1 = 550$ and $N_2 = 400$.

Common metrics to measure the performance of testing methods include the average error from the optimal solution, the sum of the regression, and the probability that the current solution is the optimal one, which is called *accuracy*[5]. For this experiment, we use the average accuracy because we know the true optimal solution beforehand. We observe whether each testing method can reach the optimal solution when it reaches the sample size bound N. After iterating the simulation 100 times for each problem and testing method, we calculate the average accuracy.

The simulation experiment result is shown in Table 4. In Problem 1, LA and our proposed methods can find the optimal solution with very high accuracy because the linear

Table 4: **Average accuracy of each testing method when the sample size bound N is elapsed.**

	Baseline			Proposal	
Problem	BF	LA	LS	LALS	LALS+
1	0.24	1.00	0.00	1.00	1.00
2	0.54	1.00	0.01	1.00	1.00
3	0.26	0.14	0.01	0.22	0.22
4	0.46	0.26	0.02	0.33	0.68

assumption device performed well with the linear function $f_1(\boldsymbol{x})$. However, BF retained low accuracy because of the variance of the observed values. LS also remained low because it cannot conduct local search. A similar tendency is observed for Problem 2. In Problem 3, the accuracy of the proposed methods dropped to the same level as the baseline accuracy because the linear assumption device did not work well with the nonlinear evaluation function $f_2(\boldsymbol{x})$. However, if the given sample size bound N is sufficiently large to conduct local search as shown in Problem 4, then our proposed methods can gain accuracy even if the evaluation function is a nonlinear function. Especially, LALS+ marked high accuracy because of the flexible allocation device. This result demonstrates to us that our proposed methods can find the optimal solution with high accuracy, except for the case in which the evaluation function is nonlinear and the provided sample size is insufficient to conduct a local search.

6.3 Simulation Experiment on an Actual Large-scale Website

Our proposed method can be confirmed to work on actual large-scale websites with a simulation experiment. Results show that our proposed methods achieve high accuracy with fewer samples than those with baseline methods. In this experiment, we design the evaluation function based on data collected from an actual large-scale website named SPYSEE[2] (hereinafter designated as "*Website L*"). Website L is a search engine website specialized to human profiles and relations. Several tens of thousands of users visit this website every day. This website includes Profile and Network pages. Profile pages display profiles of people. Network pages display the human relation networks surrounding the given person. We use this website for experimentation because of its simple structure, which enables us to set the key metrics to optimize. We apply CTR of the advertisements, which are located around the human-network section on the Network page as the key metrics to optimize.

We applied three changes (Changes A, B, C) on Network page and observed user behavior for 18 days starting from May 14, 2013. Results show that Change A improved CTR by 1.17% in average, although the others had negative impacts on the metrics. The performance of each change is presented in Table 5. The 95% confidence intervals show that each change has a significant effect.

Therefore, CTR can be expressed as the following function $q(\boldsymbol{x})$, where the variable $x_i \in \{0, 1\}$ denotes whether the change is applied ($x_i = 1$) or not ($x_i = 0$).

$$q(\boldsymbol{x}) = 0.0640 + 0.0117x_A - 0.0067x_B - 0.0134x_C$$

[2]SPYSEE `http://spysee.jp/`

Table 5: **Performance of each layout on the Network page.**

Change	Pageviews	CTR (95% confidence interval)
A	9,739	7.57% (7.04%, 8.09%)
B	14,923	5.73% (5.36%, 6.10%)
C	5,869	5.06% (4.50%, 5.62%)
Original	71,457	6.40% (6.22%, 6.58%)

We assume this possibility function as a linear function because we are collecting data from the actual website. It is difficult to repeat the experiment, which might cause crucially important damage to revenues. The evaluation function $f(\boldsymbol{x})$ returns 1, which denotes a click with probability of $q(\boldsymbol{x})$, otherwise 0. We set the parameters as $\alpha = 0.05, 1 - \beta = 0.8$, and $\Delta = 0.05$. Then the sample sizes are calculated as $N_1 = 5500$ and $N_2 = 6500$. We iterated the simulation 300 times and calculated the average accuracy for each testing method.

We applied the testing methods to the given evaluation function and obtained the results presented in Figure 4. Results show that BF improves the accuracy constantly along with the number of samples. Furthermore, LA exhibits the best performance all the time because the linear assumption performs very well with the given linear evaluation function. In addition, LS steadily improves the accuracy as solution moves are conducted, but it remains inferior to all other testing methods. Our proposed methods mark high accuracy at the end of the initialization phase, similarly to LA because the linear assumption performs well. In the local search phase, LALS improves the accuracy because it conducts a solution move and marks high accuracy compared to BF and LS. LALS+ improves the accuracy faster than LALS because the flexible allocation updates the optimal solution frequently.

6.4 Practical Experiment on an Actual Small-scale Website

To evaluate that our proposed method functions well for small-scale websites, we conduct an experiment using a real website: Imagerous[3] (hereinafter designated as "*Website S*"), which hundreds of users visit every day. We implement our proposed methods as an optimizer program and introduce it to Website S to optimize the website actually. The result demonstrates that our proposed method can find higher-performing variation in the same amount of users compared to the baseline method.

Website S is an image gallery website that lists background images for desktops and smart phones. This website consists of pages of two kinds: Album and Picture. Visitors explore a user's favorite picture on an Album page and download it from each Picture page. We use this website because the page structure is simple. It is easy to set the key metrics to optimize. We set the page views per session as the key metrics to maximize. We exclude those sessions which have hundreds of page views assuming that the visitor is not a human but a robot. We show the elements and their values on the Album page for optimization in Table 6. We express each solution by the combination of the index of each element's value. For example, the solution

[3]Imagerous `http://imagero.us/`

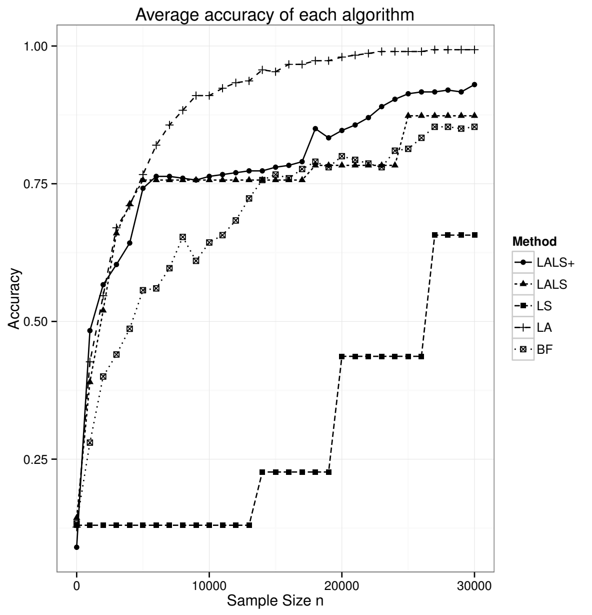

Figure 4: Accuracy of testing methods used for the evaluation function estimated from Website L.

Table 6: Elements and their values on the Album page of Website S.

Variable	Element	Values
x_1	Width of thumbnail border	0px, 5px
x_2	Margin of thumbnail	0px, 5px, 10px
x_3	Size of thumbnail	100px, 200px, 300px
x_4	Shape of thumbnail	Square, Circle

$\boldsymbol{x} = (0, 2, 1, 0)$ denotes the following settings. {Width of thumbnail border: 0 px, Margin of thumbnail: 10 px, Size of thumbnail: 200 px, Shape of thumbnail: Square}.

We apply two testing methods to Website S in this experiment: LS and LALS. We limited the testing methods to shorten the whole experiment term and to prevent extraneous factors from becoming included in the result. We expect that LALS+ will exhibit superior performance to that of LALS because a flexible allocation device is applied to them. We took LS, which corresponds to popular A/B testing, as the baseline method.

The optimizer tool separates the visitors randomly into two groups. Each group is assigned to each of the testing methods. It optimizes the website individually. We set the constants as follows: significance level $\alpha = 0.05$, statistical power $1-\beta = 0.8$, and effect size $\Delta = 0.3$. The sample sizes are calculated as $N_1 = 120$ and $N_2 = 240$.

We show the outline of the optimizer program in Figure 5. When a user accesses Website S, the browser receives both an HTML file and a javascript file from the web server. The javascript file is executed when it is loaded. It sends requests to the server specialized for website optimization (hereinafter designated as "*Optimizer server*"). Optimizer server assigns a variation to the user based on a given testing method. The assigned variation is stored in the cookie to maintain the consistency of the user experience when the user visits the website again later. The variation is described by the form of the pairs of the DOM selector and the value for the given DOM element's style attribute. The javascript program rewrites HTML code based on the variation information and shows the user the assigned variation. It also tracks user behavior by sending the activity log to Optimizer server when important events are fired, which include landing and clicking. Optimizer server runs a batch program to calculate the expected value of each solution and to update the candidate solutions every 30 min. We can run the program every time a new sample comes, but we produced it as a batch program to avoid delaying the response speed, which would degrade the user experience.

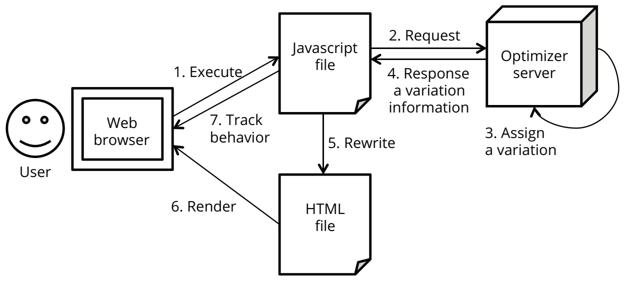

Figure 5: Outline of the optimizer program.

Table 7: Transition of the current solution and the expectation for each testing method.

Method	Sample size n	Current solution $\boldsymbol{x}$	Expectation $\mathbb{E}[y\|\boldsymbol{x}]$	Standard error
LS	0	$(1, 1, 2, 1)$	N/A	N/A
	288	$(1, 2, 2, 1)$	2.560	0.188
	593	$(1, 2, 2, 1)$	2.470	0.136
LALS	0	N/A	N/A	N/A
	122	$(1, 0, 0, 0)$	7.000	Inf.
	376	$(0, 0, 0, 0)$	3.760	0.300
	610	$(0, 0, 0, 0)$	3.873	0.214

Table 7 presents results of the experiment we ran for 14 days starting from February 14, 2014. The result shows that LALS reached a 57% higher expected value compared to LS, with an equal number of users. The difference between the two expected values was significant with 99% confidence, according to the Student t-test result. Actually, LALS started local search from the estimated promising solution, although LS was unable to improve the expected value so much from the initial solution, which was selected randomly. Therefore, our proposed method functions as a practical optimizer program on the real small-scale website and reaches higher performing variation with fewer users than those of the baseline method.

7. DISCUSSION

The experiment results demonstrate that our proposed methods reach higher performing variation with fewer users than those of baseline methods. This characteristic helps us to conduct online controlled experiments on small-scale websites without extending the experimental period, by which we can prevent seasonal factors to be included in the experiment result. Our proposed method is also robust to the increase of the solution space size. The minimum sample size for initialization phase $N_1 = \sum_{i=1}^{m} l_i$ increases linearly with the number of elements to optimize m, whereas the candidate solution size $|X| = \prod_{i=1}^{m} l_i$ increases exponentially. Therefore, we can apply our proposed method even if the given sample size bound N is less than the number of candidates.

When the evaluation function is nonlinear, our proposed method has the important shortcoming that it cannot find the optimal solution in the initialization phase. However, it can explore the optimal solution by local search phase if sufficient samples are given. In addition, our proposed methods can be developed by the application of meta-heuristic algorithms such as simulated annealing and genetic algorithm to explore global optimal solutions after the initialization phase.

One advantage of our proposed method is that it requires no parameter tuning. Meta-heuristic algorithms are powerful solutions to ascertain an approximate solution quickly from a large solution space, but the performance tends to depend on the parameter values. It is necessary to find a good value set through trial and error. However, our proposed method requires parameters only for statistical tests; the standard values for them were proposed in past studies[11]. Therefore, our proposed method is applicable to widely various websites, for which it can provide benefits quickly.

We assumed for this study that website optimization is conducted on a single web page, but our proposal is applicable to an entire website by including backend algorithms as elements to optimize. We further assumed that the elements are extracted manually, but this process can be automated through future development. Web page segmentation technology has been studied for many years. Elements have been extracted automatically from the given website[23]. The candidate values of each element can be given by implementing knowledge to the system. For example, if the given element's property is color, then we can use the knowledge of color space and make the optimizer program choose one value from it.

We demonstrated that our proposed method is valid irrespective of the scale of the website by applying the method on both large-scale Website L and small-scale Website S. We further showed that this method is applicable to multiple types of the key metrics by maximizing CTR on Website L and the page views per session on Website S. These results demonstrate that our proposed method is applicable irrespective of the website scale and the key metrics to be optimized.

The linear assumption will work well even if numerous categorical variables are included because this device merely compares the expected value of each candidate value and chooses the best one. The order of the candidate values need not be defined in this phase. However, the performance of the local search is expected to vary because the neighbor solution space will change according to the definition of the neighbor solution.

The difficulty of website optimization depends on the design of the key metrics and the experimental period. Metrics with high variance are difficult because they decrease the power of statistical hypothesis testing[19]. Long-term key metrics such as the user retention rate are also difficult to optimize because they take a long time for the effects of the modifications to be reflected in the metrics. The experimental period is also important because the website performance can be affected by the period of time such as day or night, weekday or weekend. Not only seasonal changes but also environmental changes such as the increase of smart phone users are crucially important. We must design key metrics and experimental terms to remove these unexpected effects from the experiment results.

We discuss the relation between the fractional factorial design and linear assumption device used in our proposed method. Fractional factorial design was used in traditional industries such as manufacturing and agriculture, which entails huge costs to reconfigure the environment parameters. For example, the temperature and the engine speed of the production line are crucially important to maintain the product quality, but it takes hours to stabilize them once they are changed[13]. For that reason, the experiment must be designed a priori to reduce the number of solutions to be tested. However, websites require no cost to change the parameters if they are coded to be so. We need not fix the candidates beforehand. We can merely choose the solution for random observation. Moreover, it is possible to estimate the optimal solution by conducting ANOVA after we collect samples. The traditional orthogonal array is inflexible. It has some limitations in the number of parameters and values, but we need not adjust our experiment to satisfy it as long as the reconfiguration cost is zero.

We can recognize any given problem as a website optimization problem if it satisfies the following conditions.

1. Each solution is expressed as a combination of elements.
2. Reconfiguration cost is zero.
3. User feedback is observable.

For example, manufactured goods can be recognized as a website optimization problem. The rise of 3D printing technology facilitates fabrication of numerous variations of products. We can further assume that the product is a combination of elements and use the usage log data and reviews from users as observed feedback. We can improve products that are made not only of bits but also of atoms by the application of our proposed method.

8. CONCLUSION

We formalized the website optimization problem, organized existing testing methods, and proposed a new rapid testing method that can infer high-performing variation from few users. The website optimization problem is an instance of the application of a combinational optimization problem to ascertain the best-performing variation of the given website. Our proposed method is a combination of existing testing methods and devices for efficient search processes. This method consists of two phases: an initialization phase and a local search phase. The initialization phase estimates the optimal solution assuming that the evaluation function is linear. Local search phase explores the optimal solution starting from the initial solution estimated in the initialization phase. The proposed method was evaluated using simulation experiments and practical experiments conducted on small-scale and large-scale websites. Results show that, compared to existing methods, our proposed method can reach a higher-performing solution with fewer samples. Website optimization is applicable not only to websites but also to manufactured goods.

9. REFERENCES

[1] B. Adenso-Diaz and M. Laguna. Fine-tuning of algorithms using fractional experimental designs and local search. *Operations Research*, 54(1):99–114, 2006.

[2] D. Agarwal, B.-C. Chen, and P. Elango. Fast online learning through offline initialization for time-sensitive recommendation. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 703–712. ACM, 2010.

[3] D. Agarwal, B.-C. Chen, P. Elango, and X. Wang. Click shaping to optimize multiple objectives. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 132–140. ACM, 2011.

[4] A. Asllani and A. Lari. Using genetic algorithm for dynamic and multiple criteria web-site optimizations. *European Journal of Operational Research*, 176(3):1767–1777, 2007.

[5] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002.

[6] E. Bakshy and D. Eckles. Uncertainty in online experiments with dependent data: An evaluation of bootstrap methods. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1303–1311. ACM, 2013.

[7] E. Bakshy, D. Eckles, and M. S. Bernstein. Designing and deploying online field experiments. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 283–292. International World Wide Web Conferences Steering Committee, 2014.

[8] S. Bao, G. Xue, X. Wu, Y. Yu, B. Fei, and Z. Su. Optimizing web search using social annotations. In *Proceedings of the 16th International Conference on World Wide Web*, pages 501–510. ACM, 2007.

[9] M. Birattari, T. Stützle, L. Paquete, and K. Varrentrapp. A racing algorithm for configuring metaheuristics. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 11–18, 2002.

[10] D. Chakrabarti, D. Agarwal, and V. Josifovski. Contextual advertising by combining relevance with click feedback. In *Proceedings of the 17th International Conference on World Wide Web*, pages 417–426. ACM, 2008.

[11] J. Cohen. *Statistical power analysis for the behavioral sciences*. Psychology Press, 1988.

[12] S. P. Coy, B. L. Golden, G. C. Runger, and E. A. Wasil. Using experimental design to find effective parameter settings for heuristics. *Journal of Heuristics*, 7(1):77–97, 2001.

[13] T. Crook, B. Frasca, R. Kohavi, and R. Longbotham. Seven pitfalls to avoid when running controlled experiments on the web. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1105–1114. ACM, 2009.

[14] A. Deng, Y. Xu, R. Kohavi, and T. Walker. Improving the sensitivity of online controlled experiments by utilizing pre-experiment data. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, pages 123–132. ACM, 2013.

[15] V. Heidrich-Meisner and C. Igel. Hoeffding and bernstein races for selecting policies in evolutionary direct policy search. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 401–408. ACM, 2009.

[16] J. Hu, H.-J. Zeng, H. Li, C. Niu, and Z. Chen. Demographic prediction based on user's browsing behavior. In *Proceedings of the 16th International Conference on World Wide Web*, pages 151–160. ACM, 2007.

[17] R. Kohavi. Mining e-commerce data: The good, the bad, and the ugly. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 8–13. ACM, 2001.

[18] R. Kohavi, A. Deng, R. Longbotham, and Y. Xu. Seven rules of thumb for web site experimenters. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1857–1866. ACM, 2014.

[19] R. Kohavi, R. Longbotham, D. Sommerfield, and R. M. Henne. Controlled experiments on the web: survey and practical guide. *Data Mining and Knowledge Discovery*, 18(1):140–181, 2009.

[20] L. Li, W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International Conference on World Wide Web*, pages 661–670. ACM, 2010.

[21] W. Li, X. Wang, R. Zhang, Y. Cui, J. Mao, and R. Jin. Exploitation and exploration in a performance based contextual advertising system. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 27–36. ACM, 2010.

[22] D. Sculley, R. G. Malkin, S. Basu, and R. J. Bayardo. Predicting bounce rates in sponsored search advertisements. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1325–1334. ACM, 2009.

[23] S. H. Sengamedu and R. R. Mehta. Web page layout optimization using section importance. In *Proceedings of the 17th International Conference on World Wide Web*. ACM, 2008.

[24] D. Tang, A. Agarwal, D. O'Brien, and M. Meyer. Overlapping experiment infrastructure: More, better, faster experimentation. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 17–26. ACM, 2010.

[25] J. White. *Bandit algorithms for website optimization*. O'Reilly Media, Inc., 2012.

[26] Q. Zhang, J. Sun, and E. Tsang. An evolutionary algorithm with guided mutation for the maximum clique problem. *Evolutionary Computation, IEEE Transactions on*, 9(2):192–200, 2005.